Cat. No. W473-E1-06

SYSMAC CJ Series

CJ2H-CPU6□-EIP
CJ2H-CPU6□
CJ2M-CPU□□

CJ2 CPU Unit Software

USER'S MANUAL

OMRON

© OMRON, 2008

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of OMRON.

No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice. Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

SYSMAC CJ Series
CJ2H-CPU6□-EIP
CJ2H-CPU6□
CJ2M-CPU□□
CJ2 CPU Unit Software

User's Manual

Revised February 2010

Introduction

Thank you for purchasing a CJ-series CJ2H-CPU6□(-EIP) or CJ2M-CPU□□ Programmable Controller. This manual contains information required to use the CJ2H-CPU6□(-EIP) or CJ2M-CPU□□. Please thoroughly read and understand this manual before you use the CJ2H-CPU6□(-EIP) or CJ2M-CPU□□.

Intended Audience

This manual is intended for the following personnel, who must also have knowledge of electrical systems (an electrical engineer or the equivalent).

- Personnel in charge of installing FA systems
- Personnel in charge of designing FA systems.
- Personnel in charge of managing FA systems and facilities.

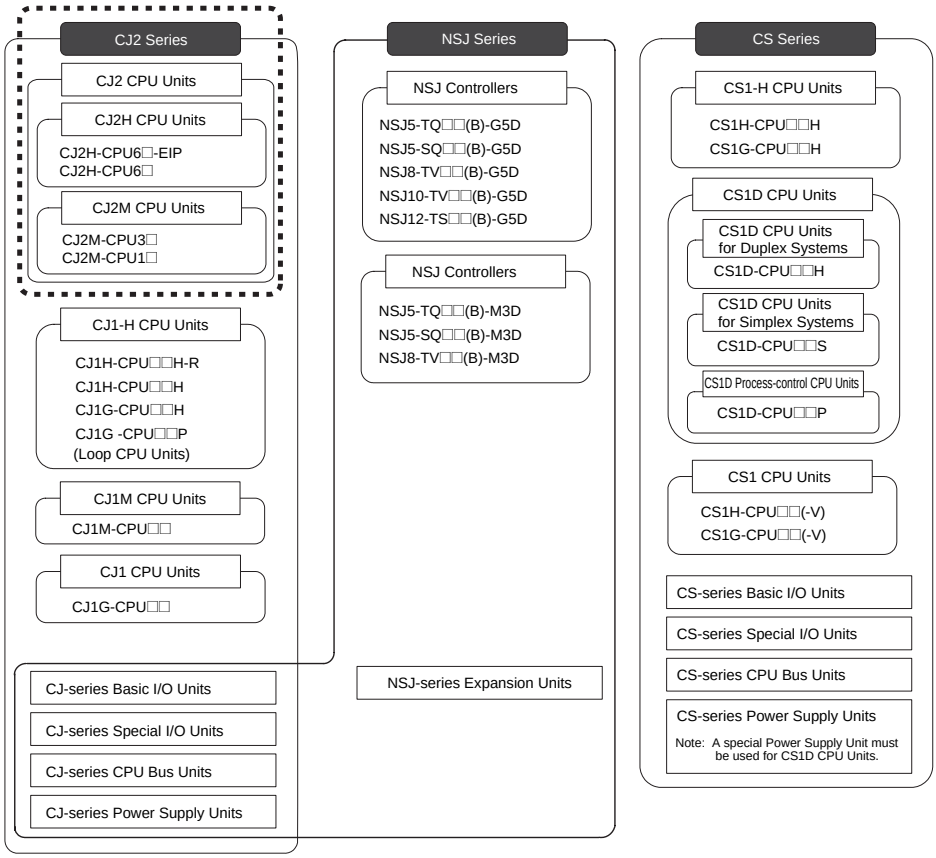
Applicable Products

CJ-series CJ2 CPU Units

- CJ2H-CPU6□-EIP
- CJ2H-CPU6□
- CJ2M-CPU3□
- CJ2M-CPU1□

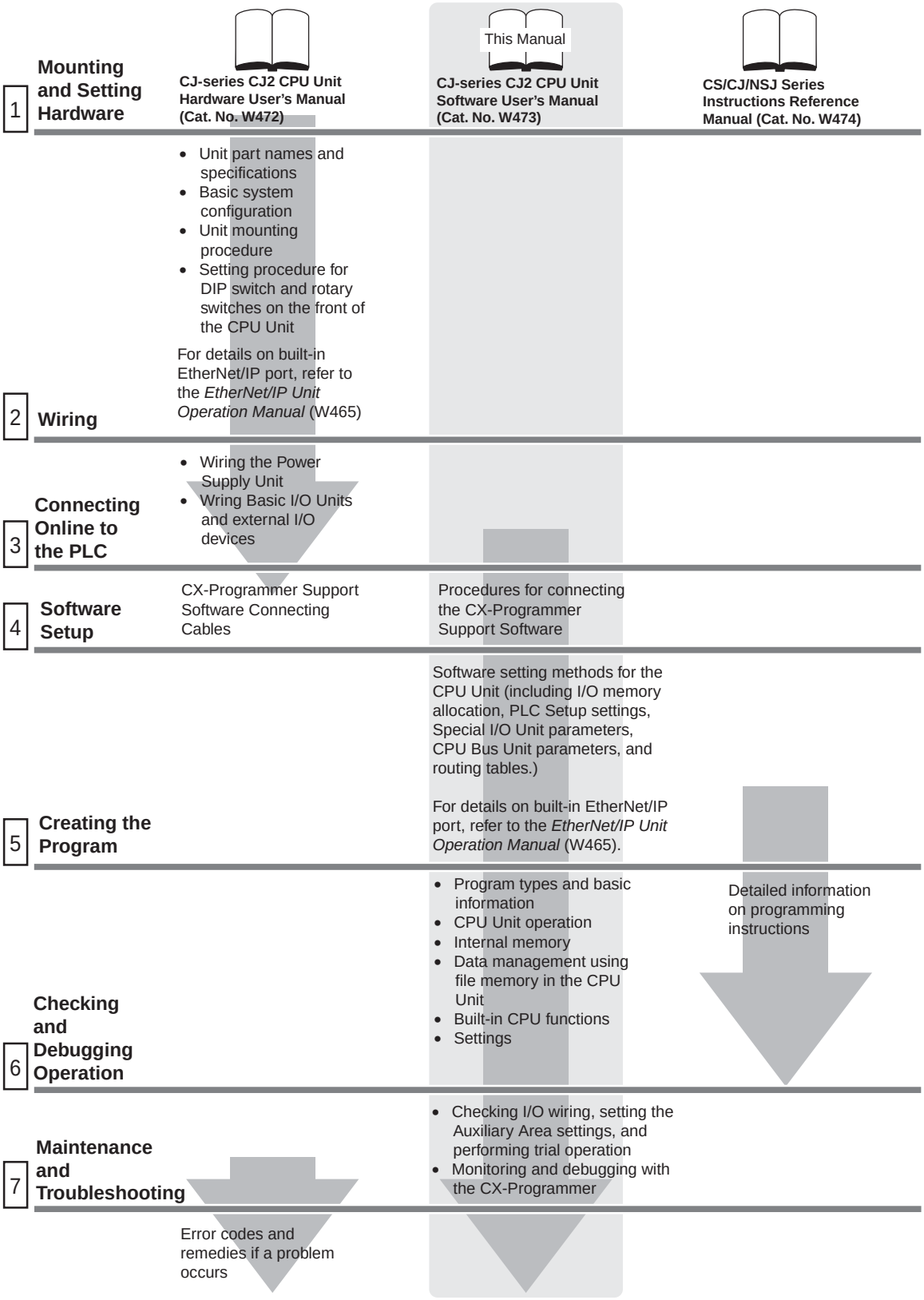
In this manual, the following notation is used to indicate the CPU Units.

- CJ2H-CPU6□(-EIP): Indicates the CJ2H-CPU6□-EIP and CJ2H-CPU6□ CPU Units.
- CJ2M-CPU□□: Indicates the CJ2M-CPU3□ and CJ2M-CPU1□ CPU Units.



CJ2 CPU Unit Manuals

Information on the CJ2 CPU Units is provided in the following manuals. Refer to the appropriate manual for the information that is required.



Manual Configuration

The CJ2 CPU manuals are organized in the sections listed in the following tables. Refer to the appropriate section in the manuals as required.

Hardware User's Manual (Cat. No. W472)

Section	Content
Section 1 Overview	This section gives an overview of the CJ2 CPU Units and describes the features and specifications.
Section 2 Basic System Configuration and Devices	This section describes the system configuration for the CJ2 CPU Unit.
Section 3 Nomenclature and Functions	This section describes the part names and functions of the CPU Unit and Configuration Units.
Section 4 Support Software	This section describes the types of Support Software to use to perform programming and debugging and how to connect the PLC to the Support Software.
Section 5 Installation	This section describes the installation locations and how to wire CPU Units and Configuration Units.
Section 6 Troubleshooting	This section describes how to check the status for errors that occur during system operation and the remedies for those errors.
Section 7 Inspection and Maintenance	This section describes periodic inspection, the service life of the Battery and Power Supply Unit, and how to replace the Battery.
Section 8 Backup Operations	This section describes the procedure to back up PLC data.
Appendices	The appendices provide Unit dimensions, details on fatal and non-fatal errors, information on connecting to serial ports on the CPU Unit, the procedure for installing the USB driver on a computer, and information on load short-circuit protection and line disconnection detection.

Software User's Manual (Cat. No. W473) (This Manual)

Section	Content
Section 1 Overview	This section gives an overview of the CJ2 CPU Units and describes the features and specifications.
Section 2 Internal Memory in the CPU Unit	This section describes the types of memory in the CPU Unit and the data that is stored.
Section 3 CPU Unit Operation	This section describes the internal operation of the CPU Unit.
Section 4 CPU Unit Initialization	This section describes the initial setup of the CPU Unit.
Section 5 Understanding Programming	This section describes program types and programming details, such as symbols and programming instructions.
Section 6 I/O Memory Areas	This section describes the I/O memory areas in the CPU Unit.
Section 7 File Operations	This section describes the files that can be stored in the CPU Unit, the storage destination for those files, and file operations.
Section 8 I/O Allocations and Unit Settings	This section describes the I/O allocations used to exchange data between the CPU Unit and other Units.
Section 9 PLC Setup	This section describes details on the PLC Setup settings, which are used to perform basic settings for the CPU Unit.
Section 10 CPU Unit Functions	This section describes functions that are built into the CPU Unit.
Section 11 Programming Devices and Communications	This section describes the procedure for connecting the CJ2 CPU Unit to the CX-Programmer or other Support Software and to other devices.
Section 12 CPU Unit Cycle Time	This section describes how to monitor and calculate the cycle time.
Appendices	The appendices provide information on programming instructions, execution times, number of steps, Auxiliary Area words and bits, a memory map of the continuous PLC memory addresses, I/O memory operation when power is interrupted, and a comparison of CJ-series and CS-series PLCs.

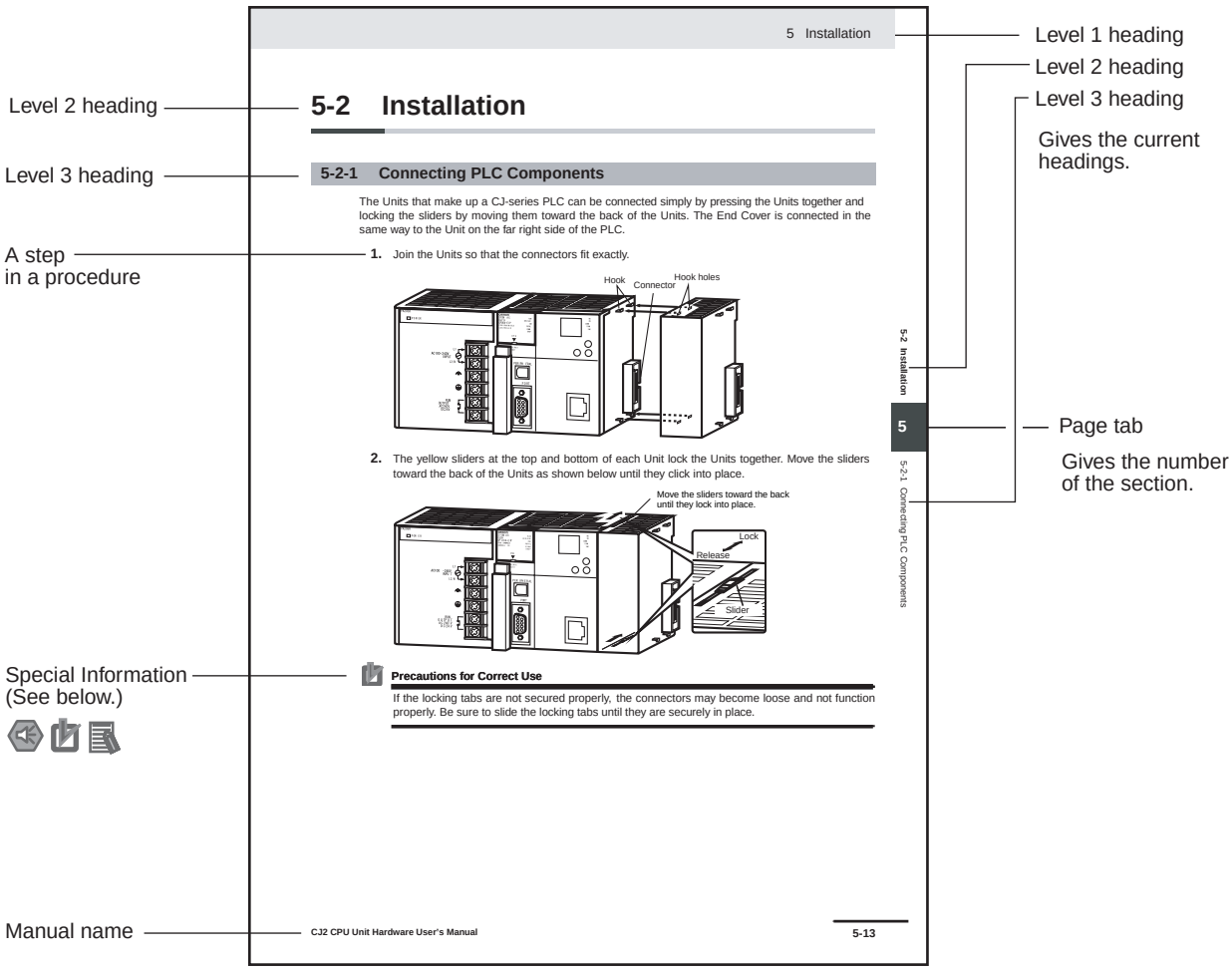
Instructions Reference Manual (Cat. No. W474)

Section	Content
Section 1 Basic Understanding of Instructions	This section provides basic information on designing ladder programs for a CS/CJ/NSJ-series CPU Unit.
Section 2 Summary of Instructions	This section provides a summary of instructions used with a CS/CJ/NSJ-series CPU Unit.
Section 3 Instructions	This section describes the functions, operands and sample programs of the instructions that are supported by a CS/CJ/NSJ-series CPU Unit.
Section 4 Instruction Execution Times and Number of Steps	This section provides the instruction execution times for each CS/CJ/NSJ-series CPU Unit instruction.
Appendices	The appendices provide a list of instructions by function code and by mnemonic and an ASCII table for the CS/CJ/NSJ-series CPU Units.

Manual Structure

Page Structure

The following page structure is used in this manual.



This illustration is provided only as a sample and may not literally appear in this manual.

Special Information

Special information in this manual is classified as follows:

Precautions for Safe Use

Precautions on what to do and what not to do to ensure using the product safely.

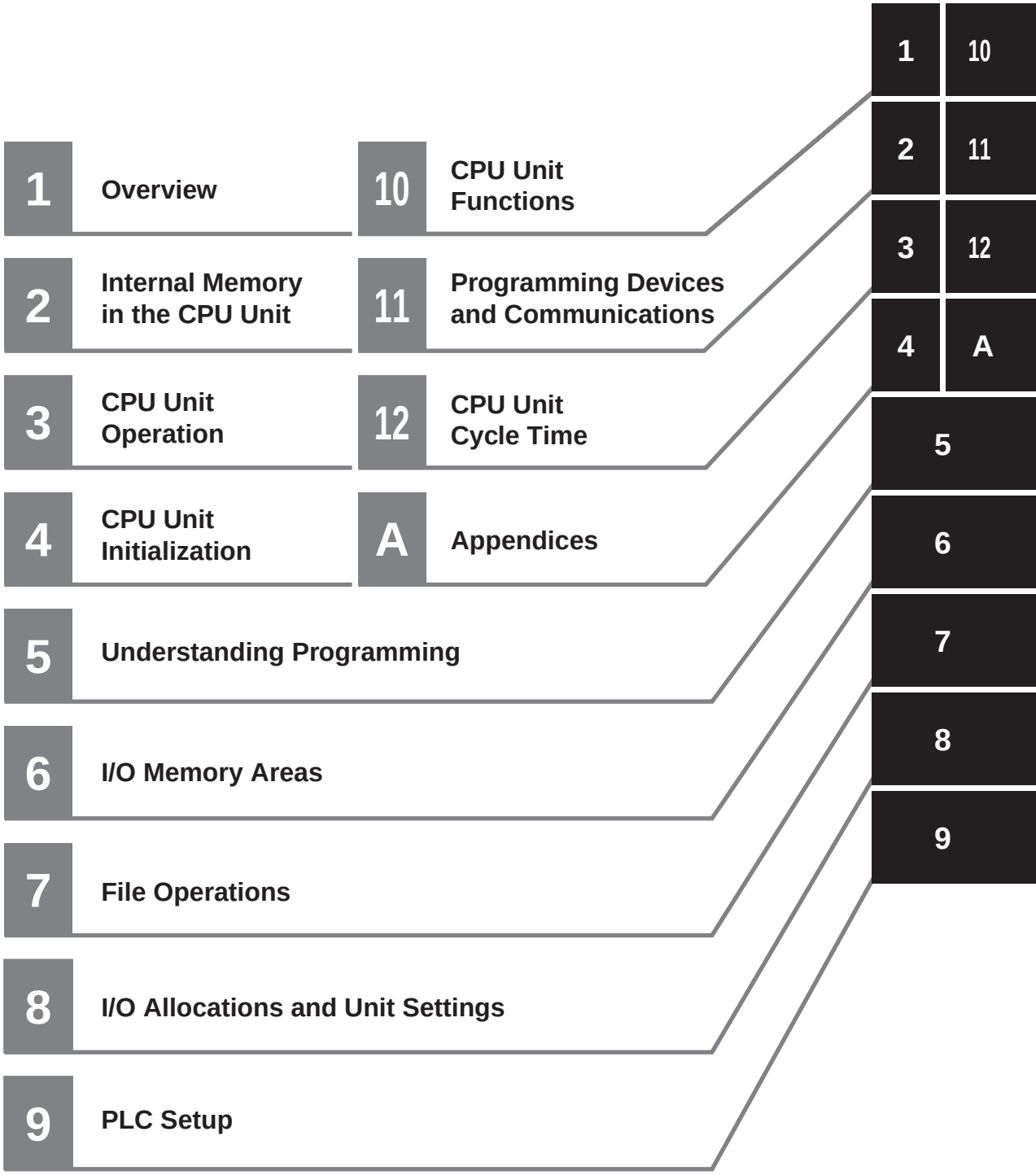
Precautions for Correct Use

Precautions on what to do and what not to do to ensure proper operation and performance.

Additional Information

Additional information to increase understanding or make operation easier.

Sections in this Manual



CONTENTS

Introduction 1

CJ2 CPU Unit Manuals 2

Manual Structure 5

Sections in this Manual 7

Safety Precautions 21

Application Precautions 25

Operating Environment Precautions 30

Regulations and Standards 31

Unit Versions of CJ2 CPU Units 33

Related Manuals 38

Section 1 Overview

1-1 Overview of CJ2 CPU Units 1-2
1-1-1 Overview 1-2
1-1-2 CJ2 CPU Unit Features 1-4

1-2 Basic Operating Procedure 1-11

Section 2 Internal Memory in the CPU Unit

2-1 Overview 2-2
2-1-1 Memory Configuration 2-2
2-1-2 Memory Areas and Stored Data 2-3
2-1-3 Transferring Data from a Programming Device to the CPU Unit 2-4

Section 3 CPU Unit Operation

3-1 CPU Unit Internal Operation 3-2
3-1-1 Overview 3-2
3-1-2 Cycle Time 3-4
3-1-3 Processing at Power Interruptions 3-7

3-2 CPU Unit Operating Modes 3-8
3-2-1 Operating Modes 3-8
3-2-2 Checking the Operating Mode 3-9
3-2-3 Changing the Operating Mode 3-10
3-2-4 Operating Mode Details 3-14

Section 4 CPU Unit Initialization

4-1 Overview of CPU Unit Initialization 4-2
4-1-1 CPU Unit Initial Settings 4-2

4-2 PLC Setup 4-8

4-3 Creating I/O Tables 4-9
4-3-1 I/O Tables 4-9
4-3-2 Automatic Allocation 4-10
4-3-3 Manual Allocation 4-10

4-4 Setting Routing Tables 4-11
4-4-1 Routing Tables 4-11
4-4-2 Cases in Which Routing Tables Are Required 4-13
4-4-3 Setting and Transferring Routing Tables 4-14

4-5 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units 4-15
4-5-1 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units 4-15
4-5-2 Setting Procedure 4-15

4-6 CPU Bus Unit Setup Area 4-16
4-6-1 CPU Bus Unit Setup Area 4-16
4-6-2 Setting Procedure 4-16

Section 5 Understanding Programming

5-1 Programming 5-3
5-1-1 Programming Overview 5-3
5-1-2 Basic Ladder Diagram Concepts 5-6
5-1-3 ST Language 5-8
5-1-4 SFC Overview 5-9

5-2 Tasks 5-11
5-2-1 Overview of Tasks 5-11
5-2-2 Cyclic Tasks 5-14
5-2-3 Interrupt Tasks 5-20
5-2-4 Designing Tasks 5-28

5-3 Sections 5-38
5-3-1 Overview of Sections 5-38

5-4 Function Blocks 5-40
5-4-1 Function Blocks 5-40
5-4-2 Features of Function Blocks 5-41
5-4-3 Function Block Specifications 5-42

5-5 Symbols 5-45
5-5-1 Overview 5-45
5-5-2 Types of Symbols 5-46
5-5-3 Global Symbols 5-48
5-5-4 Local Symbols 5-48
5-5-5 Network Symbols (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only) 5-49
5-5-6 Variables in Function Blocks 5-53
5-5-7 Symbol Data Types 5-54
5-5-8 Automatic Address Allocation to Symbols 5-59

5-6 Instructions 5-60
5-6-1 Basic Understanding of Instructions 5-60
5-6-2 Specifying Operands 5-67
5-6-3 Data Formats 5-75
5-6-4 I/O Refresh Timing 5-79

5-7 Index Registers 5-84
5-7-1 What Are Index Registers? 5-84
5-7-2 Using Index Registers 5-84
5-7-3 Processing Related to Index Registers 5-88
5-7-4 Monitoring Index Registers 5-89
5-7-5 Sharing Index and Data Registers between Tasks 5-90
5-8 Specifying Address Offsets 5-92
5-8-1 Overview 5-92
5-8-2 Examples of Address Offset Application 5-95
5-9 Checking Programs 5-96
5-9-1 Errors during CX-Programmer Input 5-96
5-9-2 Program Checks with the CX-Programmer 5-96
5-9-3 Debugging with the Simulator 5-97
5-9-4 Program Execution Check 5-100
5-10 Precautions 5-103
5-10-1 Condition Flags 5-103
5-10-2 Special Program Sections 5-108

Section 6 I/O Memory Areas

6-1 I/O Memory Areas 6-2
6-1-1 I/O Memory Area Overview 6-2
6-1-2 I/O Memory Area Structure 6-4
6-1-3 Holding I/O Memory Values 6-6
6-2 I/O Area 6-8
6-2-1 Input Bits 6-8
6-2-2 Output Bits 6-10
6-3 Data Link Area 6-13
6-4 Synchronous Data Refresh Area 6-14
6-5 CPU Bus Unit Area 6-15
6-6 Special I/O Unit Area 6-16
6-7 Serial PLC Link Area 6-17
6-8 DeviceNet Area 6-18
6-9 Work Area 6-19
6-10 Holding Area 6-20
6-11 Auxiliary Area 6-22
6-12 Temporary Relay Area 6-23
6-13 Data Memory Area 6-24
6-14 Extended Data Memory Area 6-27
6-15 Timer Areas 6-31
6-16 Counter Areas 6-33
6-17 Task Flags 6-34
6-18 Index Registers 6-35
6-19 Data Registers 6-40
6-20 Condition Flags 6-42
6-21 Clock Pulses 6-44

Section 7 File Operations

7-1 File Memory 7-2
7-1-1 Types of File Memory 7-2
7-1-2 Initializing File Memory 7-3
7-1-3 Memory Card Precautions 7-5

7-2 Types of Files Stored in File Memory 7-7
7-2-1 File Types 7-7
7-2-2 Creating and Saving Files for File Memory 7-10

7-3 File Memory Operations 7-11
7-3-1 Types of File Memory Operations 7-11
7-3-2 File Memory Operating Procedures and File Memory Files 7-13
7-3-3 Restrictions on File Use 7-19
7-3-4 File Sizes 7-20
7-3-5 Relation between Support Software and File Memory Files 7-21

Section 8 I/O Allocations and Unit Settings

8-1 I/O Allocations 8-2
8-1-1 I/O Allocations 8-2
8-1-2 Automatic Allocation 8-5
8-1-3 Manual Allocation 8-9
8-1-4 I/O Table Errors and Precautions 8-17

8-2 Setting CPU Bus Units and Special I/O Units 8-20
8-2-1 Setting Parameters 8-20
8-2-2 Data Exchange 8-24

Section 9 PLC Setup

9-1 Overview of the PLC Setup 9-2

9-2 PLC Setup Settings 9-4

9-3 PLC Setup Settings 9-5
9-3-1 Startup Operation Settings 9-5
9-3-2 CPU Unit Settings 9-8
9-3-3 Timings/Synchronous Settings 9-14
9-3-4 Special I/O Unit Cyclic Refreshing 9-19
9-3-5 Basic I/O Unit Rack Response Times 9-21
9-3-6 Serial Port Settings 9-23
9-3-7 Peripheral Service 9-30
9-3-8 FINS Protection 9-31

Section 10 CPU Unit Functions

10-1 Clock Functions 10-3
10-1-1 Clock Functions 10-3
10-1-2 Times Stored in Memory 10-4
10-1-3 Free-running Timers 10-6

10-2 Cycle Time/High-speed Processing 10-7
10-2-1 Minimum Cycle Time 10-7
10-2-2 Maximum Cycle Time 10-8
10-2-3 Monitoring the Cycle Time 10-9
10-2-4 High-speed Inputs 10-9
10-2-5 Background Execution 10-10
10-2-6 High-speed Interrupt Function 10-19

10-3 Startup Settings and Maintenance 10-22
10-3-1 Holding Settings for Operating Mode Changes and at Startup 10-22
10-3-2 Power OFF Detection Delay Setting 10-24
10-3-3 Disabling Power OFF Interrupts 10-25
10-3-4 RUN Output 10-26
10-3-5 Automatic Transfer at Startup 10-27
10-4 Unit Management Functions 10-35
10-4-1 Basic I/O Unit Management 10-35
10-4-2 CPU Bus Unit Flags/Bits 10-37
10-4-3 Special I/O Unit Flags/Bits 10-38
10-5 Memory Management Functions 10-39
10-5-1 Automatic Backup 10-39
10-5-2 EM File Memory Functions 10-41
10-5-3 Comment Memory 10-42
10-5-4 Replacing the Entire Program during Operation 10-43
10-6 Security Functions 10-50
10-6-1 Write-protection Using the DIP Switch 10-50
10-6-2 Read Protection Using Passwords 10-50
10-6-3 Program Operation Protection Using Production Lot Numbers 10-55
10-6-4 Write Protection from FINS Commands 10-56
10-6-5 PLC Names 10-60
10-7 Debugging 10-63
10-7-1 Forced Set/Reset 10-63
10-7-2 Test Input 10-64
10-7-3 Differential Monitoring 10-64
10-7-4 Online Editing 10-65
10-7-5 Turning OFF Outputs 10-67
10-7-6 Tracing Data 10-68
10-7-7 Storing the Stop Position at Errors 10-73
10-7-8 Failure Alarm Instructions 10-74
10-7-9 Simulating System Errors 10-75
10-7-10 Failure Point Detection 10-76
10-8 Synchronous Unit Operation 10-78
10-8-1 Overview 10-78
10-8-2 Details on Synchronous Unit Operation 10-81
10-8-3 Synchronous Unit Operation Specifications 10-84
10-8-4 Synchronous Data Refresh 10-85
10-8-5 Restrictions in Using Synchronous Unit Operation 10-89
10-8-6 Application Procedure 10-91
10-8-7 PLC Setup 10-92
10-8-8 Writing the Synchronous Interrupt Task 10-94
10-8-9 Adjusting and Troubleshooting Synchronous Unit Operation 10-95

Section 11 Programming Devices and Communications

11-1 Accessing a PLC from the CX-Programmer 11-2
11-1-1 Overview 11-2
11-1-2 System Configurations for Accessible PLCs 11-4
11-1-3 Accessing a PLC from the CX-Programmer 11-8
11-1-4 Automatic Online Connection 11-11
11-2 Serial Communications 11-15
11-2-1 Overview of Serial Communications 11-15
11-3 Communications Networks 11-29

Section 12 CPU Unit Cycle Time

12-1 Monitoring the Cycle Time 12-2
12-1-1 Monitoring the Cycle Time 12-2

12-2 Computing the Cycle Time 12-4
12-2-1 CPU Unit Operation Flowchart 12-4
12-2-2 Cycle Time Overview 12-5
12-2-3 I/O Unit Refresh Times for Individual Units 12-7
12-2-4 Cycle Time Calculation Example 12-11
12-2-5 Online Editing Cycle Time Extension 12-13
12-2-6 I/O Response Time 12-13
12-2-7 Response Time for Built-in Input Interrupts 12-14
12-2-8 Response Performance of Serial PLC Links 12-15

Appendices

A-1 Instruction Functions A-3
A-1-1 Sequence Input Instructions A-3
A-1-2 Sequence Output Instructions A-5
A-1-3 Sequence Control Instructions A-6
A-1-4 Timer and Counter Instructions A-10
A-1-5 Comparison Instructions A-14
A-1-6 Data Movement Instructions A-18
A-1-7 Data Shift Instructions A-20
A-1-8 Increment/Decrement Instructions A-24
A-1-9 Symbol Math Instructions A-24
A-1-10 Conversion Instructions A-29
A-1-11 Logic Instructions A-35
A-1-12 Special Math Instructions A-37
A-1-13 Floating-point Math Instructions A-38
A-1-14 Double-precision Floating-point Instructions A-42
A-1-15 Table Data Processing Instructions A-45
A-1-16 Tracking Instructions A-49
A-1-17 Data Control Instructions A-50
A-1-18 Subroutine Instructions A-54
A-1-19 Interrupt Control Instructions A-55
A-1-20 Step Instructions A-56
A-1-21 Basic I/O Unit Instructions A-56
A-1-22 Serial Communications Instructions A-59
A-1-23 Network Instructions A-61
A-1-24 File Memory Instructions A-63
A-1-25 Display Instructions A-64
A-1-26 Clock Instructions A-65
A-1-27 Debugging Instructions A-66
A-1-28 Failure Diagnosis Instructions A-66
A-1-29 Other Instructions A-67
A-1-30 Block Programming Instructions A-68
A-1-31 Text String Processing Instructions A-72
A-1-32 Task Control Instructions A-75
A-1-33 Model Conversion Instructions A-75
A-1-34 Special Function Block Instructions A-76

A-2 Instruction Execution Times and Number of Steps A-78
A-2-1 Sequence Input Instructions A-79
A-2-2 Sequence Output Instructions A-79
A-2-3 Sequence Control Instructions A-80
A-2-4 Timer and Counter Instructions A-81
A-2-5 Comparison Instructions A-82
A-2-6 Data Movement Instructions A-83
A-2-7 Data Shift Instructions A-84
A-2-8 Increment/Decrement Instructions A-85
A-2-9 Symbol Math Instructions A-85
A-2-10 Conversion Instructions A-87
A-2-11 Logic Instructions A-89
A-2-12 Special Math Instructions A-89
A-2-13 Floating-point Math Instructions A-89
A-2-14 Double-precision Floating-point Instructions A-91
A-2-15 Table Data Processing Instructions A-92
A-2-16 Tracking Instructions A-94
A-2-17 Data Control Instructions A-94
A-2-18 Subroutine Instructions A-95
A-2-19 Interrupt Control Instructions A-95
A-2-20 Step Instructions A-96
A-2-21 Basic I/O Unit Instructions A-96
A-2-22 Serial Communications Instructions A-97
A-2-23 Network Instructions A-98
A-2-24 File Memory Instructions A-98
A-2-25 Display Instructions A-98
A-2-26 Clock Instructions A-98
A-2-27 Debugging Instructions A-99
A-2-28 Failure Diagnosis Instructions A-99
A-2-29 Other Instructions A-100
A-2-30 Block Programming Instructions A-100
A-2-31 Text String Processing Instructions A-102
A-2-32 Task Control Instructions A-103
A-2-33 Model Conversion Instructions A-103
A-2-34 Special Function Block Instructions A-103
A-2-35 SFC Instructions A-103
A-2-36 Function Block Instance Execution Time A-104

A-3 Auxiliary Area A-106
A-3-1 Read-only Area (Set by System) A-106
A-3-2 Read/Write Area (Set by User) A-129
A-3-3 Details on Auxiliary Area Operation A-138

A-4 Memory Map of PLC Memory Addresses A-141
A-4-1 PLC Memory Addresses A-141
A-4-2 Memory Map A-142

A-5 Operation for Power Interruptions A-143
A-5-1 Power OFF Operation A-143
A-5-2 Instruction Execution for Power Interruptions A-145

A-6 EtherNet/IP Connections from Windows XP (SP2 or Higher) or Windows Vista A-147
A-6-1 Changing Windows Firewall Settings A-147

A-7 PLC Comparison Charts: CJ-series and CS-series PLCs A-150

A-8 Functions Supported for Unit Versions A-154

Index Index-1

Revision History Revision-1

Read and Understand this Manual

Please read and understand this manual before using the product. Please consult your OMRON representative if you have any questions or comments.

Warranty and Limitations of Liability

WARRANTY

OMRON's exclusive warranty is that the products are free from defects in materials and workmanship for a period of one year (or other period if specified) from date of sale by OMRON.

OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, REGARDING NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR PARTICULAR PURPOSE OF THE PRODUCTS. ANY BUYER OR USER ACKNOWLEDGES THAT THE BUYER OR USER ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE. OMRON DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED.

LIMITATIONS OF LIABILITY

OMRON SHALL NOT BE RESPONSIBLE FOR SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR COMMERCIAL LOSS IN ANY WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED ON CONTRACT, WARRANTY, NEGLIGENCE, OR STRICT LIABILITY.

In no event shall the responsibility of OMRON for any act exceed the individual price of the product on which liability is asserted.

IN NO EVENT SHALL OMRON BE RESPONSIBLE FOR WARRANTY, REPAIR, OR OTHER CLAIMS REGARDING THE PRODUCTS UNLESS OMRON'S ANALYSIS CONFIRMS THAT THE PRODUCTS WERE PROPERLY HANDLED, STORED, INSTALLED, AND MAINTAINED AND NOT SUBJECT TO CONTAMINATION, ABUSE, MISUSE, OR INAPPROPRIATE MODIFICATION OR REPAIR.

Application Considerations

SUITABILITY FOR USE

OMRON shall not be responsible for conformity with any standards, codes, or regulations that apply to the combination of products in the customer's application or use of the products.

At the customer's request, OMRON will provide applicable third party certification documents identifying ratings and limitations of use that apply to the products. This information by itself is not sufficient for a complete determination of the suitability of the products in combination with the end product, machine, system, or other application or use.

The following are some examples of applications for which particular attention must be given. This is not intended to be an exhaustive list of all possible uses of the products, nor is it intended to imply that the uses listed may be suitable for the products:

- Outdoor use, uses involving potential chemical contamination or electrical interference, or conditions or uses not described in this manual.
- Nuclear energy control systems, combustion systems, railroad systems, aviation systems, medical equipment, amusement machines, vehicles, safety equipment, and installations subject to separate industry or government regulations.
- Systems, machines, and equipment that could present a risk to life or property.

Please know and observe all prohibitions of use applicable to the products.

NEVER USE THE PRODUCTS FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON PRODUCTS ARE PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

PROGRAMMABLE PRODUCTS

OMRON shall not be responsible for the user's programming of a programmable product, or any consequence thereof.

Disclaimers

CHANGE IN SPECIFICATIONS

Product specifications and accessories may be changed at any time based on improvements and other reasons.

It is our practice to change model numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the products may be changed without any notice. When in doubt, special model numbers may be assigned to fix or establish key specifications for your application on your request. Please consult with your OMRON representative at any time to confirm actual specifications of purchased products.

DIMENSIONS AND WEIGHTS

Dimensions and weights are nominal and are not to be used for manufacturing purposes, even when tolerances are shown.

PERFORMANCE DATA

Performance data given in this manual is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of OMRON's test conditions, and the users must correlate it to actual application requirements. Actual performance is subject to the OMRON Warranty and Limitations of Liability.

ERRORS AND OMISSIONS

The information in this manual has been carefully checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical, or proofreading errors, or omissions.

Safety Precautions

Definition of Precautionary Information

The following notation is used in this manual to provide precautions required to ensure safe usage of a CJ-series PLC. The safety precautions that are provided are extremely important to safety. Always read and heed the information provided in all safety precautions.

WARNING	Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury. Additionally, there may be severe property damage.
Caution	Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury, or property damage.

Precautions for Safe Use

Indicates precautions on what to do and what not to do to ensure using the product safely.

Precautions for Correct Use

Indicates precautions on what to do and what not to do to ensure proper operation and performance.

Symbols



The triangle symbol indicates precautions (including warnings). The specific operation is shown in the triangle and explained in text. This example indicates a precaution for electric shock.



The circle and slash symbol indicates operations that you must not do. The specific operation is shown in the circle and explained in text.



The filled circle symbol indicates operations that you must do. The specific operation is shown in the circle and explained in text. This example shows a general precaution for something that you must do.



The triangle symbol indicates precautions (including warnings). The specific operation is shown in the triangle and explained in text. This example indicates a general precaution.



The triangle symbol indicates precautions (including warnings). The specific operation is shown in the triangle and explained in text. This example indicates a precaution for hot surfaces.

⚠ WARNING

Do not attempt to take any Unit apart or touch the inside of any Unit while the power is being supplied. Doing so may result in electric shock.



Do not touch any of the terminals or terminal blocks while the power is being supplied. Doing so may result in electric shock.



Provide safety measures in external circuits (i.e., not in the Programmable Controller), including the following items, to ensure safety in the system if an abnormality occurs due to malfunction of the Programmable Controller or another external factor affecting the operation of the Programmable Controller. "Programmable Controller" indicates the CPU Unit and all other Units and is abbreviated "PLC" in this manual. Not doing so may result in serious accidents.



- The PLC will turn OFF all outputs when its self-diagnosis function detects any error or when a severe failure alarm (FALS) instruction is executed. Unexpected operation, however, may still occur for errors in the I/O control section, errors in I/O memory, and other errors that cannot be detected by the self-diagnosis function. As a countermeasure for all such errors, external safety measures must be provided to ensure safety in the system.
- The PLC outputs may remain ON or OFF due to deposition or burning of the output relays or destruction of the output transistors. As a countermeasure for such problems, external safety measures must be provided to ensure safety in the system.
- Provide measures in the computer system and programming to ensure safety in the overall system even if communications errors or malfunctions occur in data link communications or remote I/O communications.

Confirm safety before transferring data files stored in the file memory (Memory Card or EM file memory) to the I/O area (CIO) of the CPU Unit using a peripheral tool. Otherwise, the devices connected to the output unit may malfunction regardless of the operation mode of the CPU Unit.



Fail-safe measures must be taken by the customer to ensure safety in the event of incorrect, missing, or abnormal signals caused by broken signal lines, momentary power interruptions, or other causes. Serious accidents may result from abnormal operation if proper measures are not provided.



⚠ Caution

Execute online edit only after confirming that no adverse effects will be caused by extending the cycle time. Otherwise, the input signals may not be readable.



Confirm safety at the destination node before transferring a program, PLC Setup, I/O tables, I/O memory contents, or parameters to another node or changing contents of the any of these items. Transferring or changing data can result in unexpected system operation.



The CJ2 CPU Units automatically back up the user program and parameter data to flash memory when these are written to the CPU Unit. I/O memory including the DM, EM, and Holding Areas), however, is not written to flash memory.

The DM, EM, and Holding Areas can be held during power interruptions with a battery. If there is a battery error, the contents of these areas may not be accurate after a power interruption. If the contents of the DM, EM, and Holding Areas are used to control external outputs, prevent inappropriate outputs from being made whenever the Battery Error Flag (A402.04) is ON.



Tighten the terminal screws on the AC Power Supply Unit to the torque specified in the operation manual. The loose screws may result in burning or malfunction.



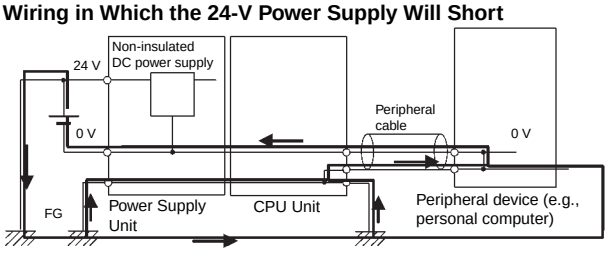
Do not touch the Power Supply Unit when power is being supplied or immediately after the power supply is turned OFF. The Power Supply Unit will be hot and you may be burned.



When connecting a personal computer or other peripheral device to a PLC to which a non-insulated Power Supply Unit (CJ1W-PD022) is mounted, either ground the 0 V side of the external power supply or do not ground the external power supply at all ground. A short-circuit will occur in the external power supply if incorrect grounding methods are used. Never ground the 24 V side, as shown below.



Wiring in Which the 24-V Power Supply Will Short



Application Precautions

Observe the following precautions when using a CJ-series PLC.

Power Supply

- Always use the power supply voltages specified in the user's manuals. An incorrect voltage may result in malfunction or burning.
- Exceeding the capacity of the Power Supply Unit may prevent the CPU Unit or other Units from starting.
- Take appropriate measures to ensure that the specified power with the rated voltage and frequency is supplied. Be particularly careful in places where the power supply is unstable. An incorrect power supply may result in malfunction.
- Always turn OFF the power supply to the PLC before attempting any of the following. Not turning OFF the power supply may result in malfunction or electric shock.
  - Mounting or dismounting Power Supply Units, I/O Units, CPU Units, Option Boards, or any other Units.
  - Assembling the Units.
  - Setting DIP switches or rotary switches.
  - Connecting cables or wiring the system.
  - Connecting or disconnecting the connectors.
- When cross-wiring terminals, the total current for all the terminal will flow in the wire. Make sure that the current capacity of the wire is sufficient.
- Observe the following precautions when using a Power Supply Unit that supports the Replacement Notification Function.
  - Replace the Power Supply Unit within six months if the display on the front of the Power Supply Unit alternates between 0.0 and A02, or if the alarm output automatically turns OFF.
  - Keep the alarm output cable separated from power line and high-voltage lines.
  - Do not apply a voltage or connect a load exceeding the specifications to the alarm output.
  - When storing the Power Supply Unit for more than three months, store it at –20 to 30°C and 25% to 70% humidity to preserve the Replacement Notification Function.
  - If the Power Supply Unit is not installed properly, heat buildup may cause the replacement notification signal to appear at the wrong time or may cause interior elements to deteriorate or become damaged. Use only the standard installation method.
- Do not touch the terminals on the Power Supply Unit immediately after turning OFF the power supply. Residual voltage may cause electrical shock.
- Observe the following precautions to prevent failure due to difference in electrical potential if the computer is connected to the PLC.
  - Before connecting a laptop computer to the PLC, disconnect the power supply plug of the computer from the AC outlet. Residual current in the AC adaptor may cause difference in electrical potential to occur between the computer and the PLC. After you connect the computer and PLC, supply the power again from the AC adaptor.
  - If the computer has an FG terminal, make the connections so that it has the same electrical potential as the FG (GR) terminal on the PLC.
- If the computer is grounded to a separate location, difference in electrical potential may occur depending on the grounding conditions.

Installation

- Do not install the PLC near sources of strong high-frequency noise.
- Before touching a Unit, be sure to first touch a grounded metallic object in order to discharge any static build-up. Not doing so may result in malfunction or damage.

- Be sure that the terminal blocks, connectors, Memory Cards, Option Boards, expansion cables, and other items with locking devices are properly locked into place. Improper locking may result in malfunction.
- The sliders on the tops and bottoms of the Power Supply Unit, CPU Unit, I/O Units, Special I/O Units, and CPU Bus Units must be completely locked (until they click into place) after connecting to adjacent Units. The Unit may not operate properly if the sliders are not locked in place. It may not be possible to achieve proper functionality if the sliders are not locked.

Wiring

- Follow the instructions in this manual to correctly perform wiring.
- Double-check all wiring and switch settings before turning ON the power supply. Incorrect wiring may result in burning.
- Be sure that all terminal screws, and cable connector screws are tightened to the torque specified in the relevant manuals. Incorrect tightening torque may result in malfunction.
- Mount terminal blocks and connectors only after checking the mounting location carefully.
- Leave the label attached to the Unit when wiring. Removing the label may result in malfunction if foreign matter enters the Unit.
- Remove the label after the completion of wiring to ensure proper heat dissipation. Leaving the label attached may result in malfunction.
- Use crimp terminals for wiring. Do not connect bare stranded wires directly to terminals. Connection of bare stranded wires may result in burning.
- Do not apply voltages to the Input Units in excess of the rated input voltage. Excess voltages may result in burning.
- Always connect to a ground of 100 Ω or less when installing the Units. Not connecting to a ground of 100 Ω or less may result in electric shock.
  A ground of 100 Ω or less must be installed when shorting the GR and LG terminals on the Power Supply Unit.
- Do not apply voltages or connect loads to the Output Units in excess of the maximum switching capacity. Excess voltage or loads may result in burning.
- Do not pull on the cables or bend the cables beyond their natural limit. Doing either of these may break the cables.
- Do not place objects on top of the cables or other wiring lines. Doing so may break the cables.
- Do not use commercially available RS-232C personal computer cables. Always use the special cables listed in this manual or make cables according to manual specifications. Using commercially available cables may damage the external devices or CPU Unit.
- Never connect pin 6 (5-V power supply) on the RS-232C port on the CPU Unit to any device other than an NT-AL001 Link Adapter, CJ1W-CIF11 Converter, and Programmable Terminals (NV3W-M□20L). The external device or the CPU Unit may be damaged.

Handling

- The Power Supply Unit may possibly be damaged if the entire voltage for a dielectric strength test is applied or shut OFF suddenly using a switch. Use a variable resistor to gradually increase and decrease the voltage.
- Separate the line ground terminal (LG) from the functional ground terminal (GR) on the Power Supply Unit before performing withstand voltage tests or insulation resistance tests. Not doing so may result in burning.
- Make sure that the DIP switches and DM Area are set correctly before starting operation.
- After replacing the CPU Unit, a Special I/O Unit, or a CPU Bus Unit, make sure that the required data for the DM Area, Holding Area, and other memory areas has been transferred to the new Unit before restarting operation.
- Confirm that no adverse effect will occur in the system before attempting any of the following. Not doing so may result in an unexpected operation.
  - Changing the operating mode of the PLC (including the setting of the startup operating mode).

- Force-setting/force-resetting any bit in memory.
- Changing the present value of any word or any set value in memory.

- Do not attempt to disassemble, repair, or modify any Units. Any attempt to do so may result in malfunction, fire, or electric shock.
- Do not drop the PLC or subject abnormal vibration or shock to it.
- The life of the battery will be reduced if the PLC is left for a period of time without a battery installed and without power supply, and then a battery is installed without turning ON the power supply.
- Replace the battery as soon as a battery error occurs or as soon as the specified battery backup time expires. Be sure to install a replacement battery within two years of the production date shown on the battery's label.
- Before replacing the battery, turn ON power for at least 5 minutes before starting the replacement procedure and complete replacing the battery within 5 minutes of turning OFF the power supply. Memory contents may be corrupted if this precaution is not obeyed.
- If the Battery Error Flag is used in programming the application, confirm system safety even if the system detects a battery error before you replace the battery while the power is ON.
- Do not short the battery terminals or charge, disassemble, heat, or incinerate the battery. Do not subject the battery to strong shocks. Doing any of these may result in leakage, rupture, heat generation, or ignition of the battery. Dispose of any battery that has been dropped on the floor or otherwise subjected to excessive shock. Batteries that have been subjected to shock may leak if they are used.
- UL standards require that only an experienced engineer can replace the battery. Make sure that an experienced engineer is in charge of battery replacement. Follow the procedure for battery replacement given in this manual.
- Dispose of the product and batteries according to local ordinances as they apply.



廢電池請回收

- If the I/O Hold Bit is turned ON, the outputs from the PLC will not be turned OFF and will maintain their previous status when the PLC is switched from RUN or MONITOR mode to PROGRAM mode. Make sure that the external loads will not produce dangerous conditions when this occurs. (When operation stops for a fatal error, including those produced with the FALS(007) instruction, all outputs from Output Unit will be turned OFF and only the internal output status will be maintained.)
- Unexpected operation may result if inappropriate data link tables or parameters are set. Even if appropriate data link tables and parameters have been set, confirm that the controlled system will not be adversely affected before starting or stopping data links.
- Write programs so that any data that is received for data link communications is used only if there are no errors in the CPU Units that are the sources of the data. Use the CPU Unit error information in the status flags to check for errors in the source CPU Units. If there are errors in source CPU Units, they may send incorrect data.
- All CPU Bus Units will be restarted when routing tables are transferred from a Programming Device to the CPU Unit. Restarting these Units is required to read and enable the new routing tables. Confirm that the system will not be adversely affected before transferring the routing tables.
- Tag data links will stop between related nodes while tag data link parameters are being transferred during PLC operation. Confirm that the system will not be adversely affected before transferring the tag data link parameters.
- If there is interference with network communications, output status will depend on the devices that are being used. When using devices with outputs, confirm the operation that will occur when there is interference with communications, and implement safety measures as required.

- When creating an AUTOEXEC.IOM file from a Programming Device (a Programming Console or the CX-Programmer) to automatically transfer data at startup, set the first write address to D20000 and be sure that the size of data written does not exceed the size of the DM Area. When the data file is read from the Memory Card at startup, data will be written in the CPU Unit starting at D20000 even if another address was set when the AUTOEXEC.IOM file was created. Also, if the DM Area is exceeded (which is possible when the CX-Programmer is used), the remaining data will be written to the EM Area.
- The user program and parameter area data in the CJ2 CPU Units are backed up in the built-in flash memory. The BKUP indicator will light on the front of the CPU Unit when the backup operation is in progress. Do not turn OFF the power supply to the CPU Unit when the BKUP indicator is lit. The data will not be backed up if power is turned OFF.
- Check the user program and Unit parameter settings for proper execution before actually running them on the Unit. Not checking the program and parameter settings may result in an unexpected operation.
- When setting a Special I/O Unit or CPU Bus Unit in the I/O tables, carefully check the safety of the devices at the connection target before restarting the Unit.
- Do not turn OFF the power supply to the PLC when reading or writing a Memory Card. Also, do not remove the Memory Card when the BUSY indicator is lit. Doing so may make the Memory Card unusable.
  To remove a Memory Card, first press the memory card power supply switch and then wait for the BUSY indicator to go out before removing the Memory Card.
- When restoring data, carefully check that the selected data is the correct data to be restored before executing the restore operation. Depending on the contents of the selected data, the control system may operate unexpectedly after the data is restored.
- Some Special I/O Units and CPU Bus Units operate with parameters stored in the CPU Unit (e.g., words allocated in DM Area, data link tables, or Ethernet settings). Information on restrictions will be displayed in the Information Area in the PLC Backup Tool if there are any restrictions for the selected CPU Bus Unit or Special I/O Unit. Check the restrictions, and then be sure to select both the CPU Unit and the CPU Bus Unit or Special I/O Unit when backing up or restoring data. The control system may operate unexpectedly if the equipment is started with the data backed up or restored without selecting both Units.
- Information on restrictions will be displayed in the Information Area in the PLC Backup Tool if the data to be stored includes a Unit that has restrictions on backup. Check the information on restrictions and take the required countermeasures. The control system may operate unexpectedly when the equipment is operated after the data is restored
- Before restoring data during PLC operation, be sure that there will be no problem if PLC operation stops. If the PLC stops at an unexpected time, the control system may operate unexpectedly.
- Be sure to turn the PLC power supply OFF and then back ON after restoring data. If the power is not reset, the system may not be updated with the restored data, and the control system may operate unexpectedly.
- Data on forced status can be backed up but it cannot be restored. Perform the procedure to force-set or force-reset bits from the CX-Programmer as required before starting operation after restoring data that includes forced status. Depending on the difference in the forced status, the control system may operate unexpectedly.
- If a symbol or memory address (only symbols are allowed for ST programming) is specified for the suffix of an array variable in ladder or ST programming, be sure that the specified element number does not exceed the maximum memory area range.
  Specifying an element number that exceeds the maximum range of the memory area specified for the symbol will result accessing data in a different memory area, and may result in unexpected operation.
- If a symbol or address is specified for an offset in a ladder diagram, program so that the memory area of the start address is not exceeded when the offset is specified indirectly using a word address or symbol.
  If an indirect specification causes the address to exceed the area of the start address, the system will access data in other area, and unexpected operation may occur.

External Circuits

- Always turn ON power to the PLC before turning ON power to the control system. If the PLC power supply is turned ON after the control power supply, temporary errors may result in control system signals because the output terminals on DC Output Units and other Units will momentarily turn ON when power is turned ON to the PLC.
- Install external breakers and take other safety measures against short-circuiting in external wiring. Insufficient safety measures against short-circuiting may result in burning.

Operating Environment Precautions

- Follow the instructions in this manual to correctly perform installation.

- Do not operate the control system in the following locations:
  - Locations subject to direct sunlight.
  - Locations subject to temperatures or humidity outside the range specified in the specifications.
  - Locations subject to condensation as the result of severe changes in temperature.
  - Locations subject to corrosive or flammable gases.
  - Locations subject to dust (especially iron dust) or salts.
  - Locations subject to exposure to water, oil, or chemicals.
  - Locations subject to shock or vibration.

- Take appropriate and sufficient countermeasures when installing systems in the following locations:
  - Locations subject to static electricity or other forms of noise.
  - Locations subject to strong electromagnetic fields.
  - Locations subject to possible exposure to radioactivity.
  - Locations close to power supplies.

Regulations and Standards

Conformance to EC Directives

Applicable Directives

- EMC Directives
- Low Voltage Directive

Concepts

● EMC Directives

OMRON devices that comply with EC Directives also conform to the related EMC standards so that they can be more easily built into other devices or the overall machine. The actual products have been checked for conformity to EMC standards (see the following note). Whether the products conform to the standards in the system used by the customer, however, must be checked by the customer.

EMC-related performance of the OMRON devices that comply with EC Directives will vary depending on the configuration, wiring, and other conditions of the equipment or control panel on which the OMRON devices are installed.

The customer must, therefore, perform the final check to confirm that devices and the overall machine conform to EMC standards.

* Applicable EMC (Electromagnetic Compatibility) standards are as follows:
EMS (Electromagnetic Susceptibility):
CS Series: EN 61131-2 and EN 61000-6-2
CJ Series: EN 61000-6-2

* EMI (Electromagnetic Interference):
EN 61000-6-4 (Radiated emission: 10-m regulations)

● Low Voltage Directive

Always ensure that devices operating at voltages of 50 to 1,000 VAC and 75 to 1,500 VDC meet the required safety standards for the PLC (EN 61131-2).

● Conformance to EC Directives

The CJ-series PLCs comply with EC Directives. To ensure that the machine or device in which the CJ-series PLC is used complies with EC Directives, the PLC must be installed as follows:

- The CJ-series PLC must be installed within a control panel.
- You must use reinforced insulation or double insulation for the DC power supplies connected to DC Power Supply Units and I/O Units.
- CJ-series PLCs complying with EC Directives also conform to the Common Emission Standard (EN 61000-6-4). Radiated emission characteristics (10-m regulations) may vary depending on the configuration of the control panel used, other devices connected to the control panel, wiring, and other conditions. You must therefore confirm that the overall machine or equipment complies with EC Directives.

Conformance to Shipbuilding Standards

This product conforms to the following shipbuilding standards. Applicability to the shipbuilding standards is based on certain usage conditions. It may not be possible to use the product in some locations. Contact your OMRON representative before attempting to use a PLC on a ship.

Usage Conditions for NK and LR Shipbuilding Standards

● Usage Conditions for Applications Other Than on the Bridge or Deck

- The PLC must be installed in a control panel.
- Gaps in the door to the control panel must be completely filled or covered with gaskets or other material.

● Usage Conditions for Bridge and Deck (Certified Only by NK)

- The PLC must be installed in a control panel.
- Gaps in the door to the control panel must be completely filled or covered with gaskets or other material.
- The following noise filter must be connected to the power supply line.

Noise Filter

Manufacturer	Cosel Co., Ltd.
Model	TAH-06-683

Trademarks

SYSMAC is a registered trademark for Programmable Controllers made by OMRON Corporation.

CX-One is a registered trademark for Programming Software made by OMRON Corporation.

Windows is a registered trademark of Microsoft Corporation.

Other system names and product names in this document are the trademarks or registered trademarks of their respective companies.

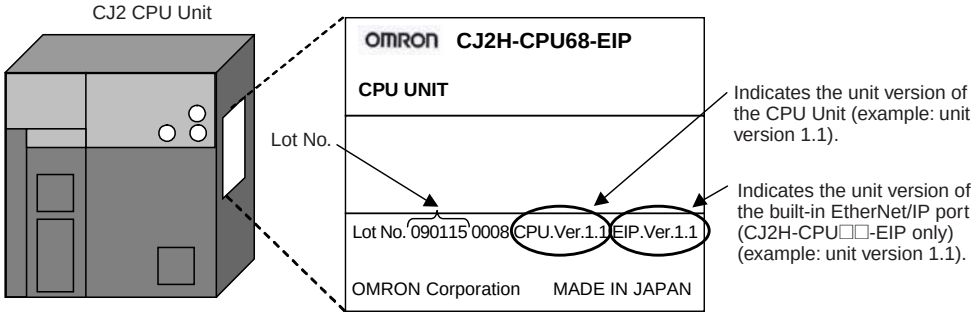
Unit Versions of CJ2 CPU Units

Unit Versions

A “unit version” has been introduced to manage CJ2 CPU Units according to differences in functionality accompanying version upgrades.

Notation of Unit Versions on Products

The unit version is given to the right of the lot number on the nameplate of the products for which unit versions are being managed, as shown below.



Confirming Unit Versions with Support Software

CX-Programmer can be used to confirm the unit version using one of the following two methods.

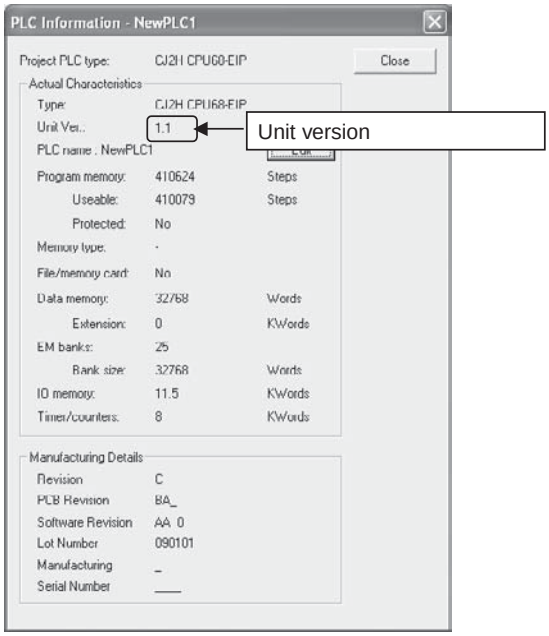
- Using the **PLC Information**
- Using the ***Unit Manufacturing Information*** (This method can be used for Special I/O Units and CPU Bus Units as well.)

● PLC Information

1 Use one of the following methods to display the PLC Information Dialog Box.

- If you know the device type and CPU type, select them in the *Change PLC* Dialog Box, go online, and select ***PLC - Edit - Information*** from the menus.
- If you don't know the device type and CPU type, but are connected directly to the CPU Unit on a serial line, select ***PLC - Auto Online*** to go online, and then select ***PLC - Edit - Information*** from the menus.

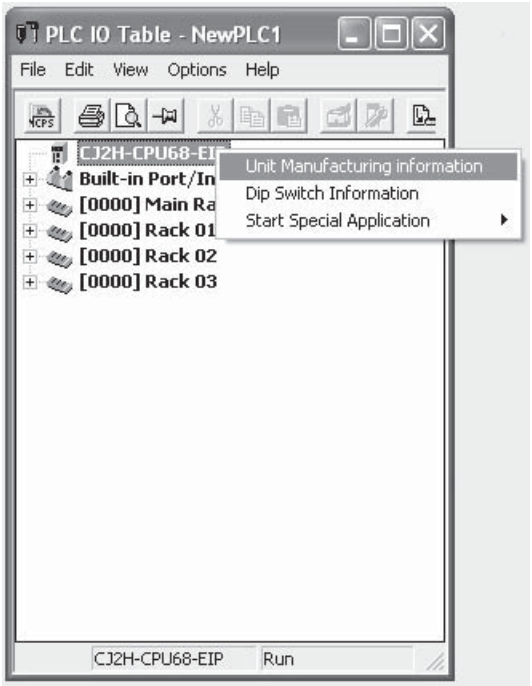
2 In either case, the following *PLC Information* Dialog Box will be displayed.



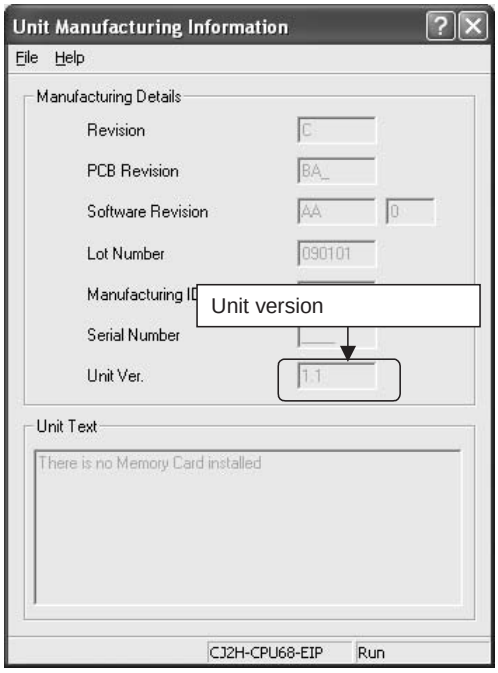
Use the above display to confirm the unit version of the CPU Unit.

● Unit Manufacturing Information

1 In the IO Table Window, right-click and select Unit Manufacturing information - CPU Unit.



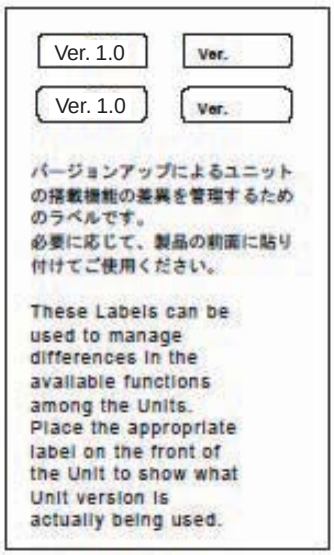
2 The following *Unit Manufacturing information* Dialog Box will be displayed.



Use the above display to confirm the unit version of the CPU Unit connected online.

3 Using the Unit Version Labels

The following unit version labels are provided with the CPU Unit.



These labels can be attached to the front of previous CPU Units to differentiate between CPU Units of different unit versions.

Unit Versions

Item	Models	Unit version
CJ2H CPU Unit	CJ2H-CPU6□-EIP	Unit version 1.0 (Built-in EtherNet/IP section: Unit version 2.0)
		Unit version 1.1 (Built-in EtherNet/IP section: Unit version 2.0)
		Unit version 1.2 (Built-in EtherNet/IP section: Unit version 2.0)
		Unit version 1.3 (Built-in EtherNet/IP section: Unit version 2.0)
	CJ2H-CPU6□	Unit version 1.1
		Unit version 1.2
		Unit version 1.3
CJ2M CPU Unit	CJ2M-CPU3□	Unit version 1.0 (Built-in EtherNet/IP section: Unit version 2.0)
	CJ2M-CPU1□	Unit version 1.0

Unit Versions and Programming Devices

The following tables show the relationship between unit versions and CX-Programmer versions. Refer to *A-8 Functions Supported for Unit Versions* for the functions supported by each unit version.

● Unit Versions and Programming Devices

CPU Unit		Functions		Required Programming Device					
				CX-Programmer					Programming Console
				Ver. 7.1 or lower	Ver. 8.0	Ver. 8.2 or higher	Ver. 9.0	Ver. 9.1 or higher	
CJ2H	CJ2H-CPU6□-EIP Unit version 1.0	Functions for unit version 1.0		---	OK	OK	OK	OK	---*3
	CJ2H-CPU6□-EIP Unit version 1.1	Functions added for unit version 1.1	Using new functions	---	---	OK*2	OK	OK	
			Not using new functions	---	OK*1	OK	OK	OK	
	CJ2H-CPU6□ Unit version 1.1	Functions added for unit version 1.1	Using new functions	---	---	OK*2	OK	OK	
			Not using new functions	---	---	OK	OK	OK	
	CJ2H-CPU6□-EIP Unit version 1.2	Functions added for unit version 1.2	Using new functions	---	---	---	OK	OK	
			Not using new functions	---	OK*1	OK*1	OK	OK	
	CJ2H-CPU6□ Unit version 1.2	Functions added for unit version 1.2	Using new functions	---	---	---	OK	OK	
			Not using new functions	---	OK*1	OK*1	OK	OK	
	CJ2H-CPU6□-EIP Unit version 1.3	Functions added for unit version 1.3	Using new functions	---	---	---	---	OK	
			Not using new functions	---	OK*1	OK*1	OK	OK	
	CJ2H-CPU6□ Unit version 1.3	Functions added for unit version 1.3	Using new functions	---	---	---	---	OK	
			Not using new functions	---	OK*1	OK*1	OK	OK	
CJ2M	CJ2M-CPU□□ Unit version 1.0	Functions for unit version 1.0		---	---	---	---	OK	

*1 It is not necessary to upgrade the version of the CX-Programmer if functionality that was enhanced for the upgrade of the CPU Unit will not be used.

*2 CX-Programmer version 8.2 or higher is required to use the functions added for unit version 1.1. The high-speed interrupt function and changing the minimum cycle time setting in MONITOR mode, however, are also supported by CX-Programmer version 8.02.

*3 A Programming Console cannot be used with a CJ2 CPU Unit.

● Pull-down List for PLC Models

Unit versions are not differentiated in the pull-down list for PLC models in the Change PLC Dialog Box of the CX-Programmer. Select as shown in the following table regardless of the unit version.

Series	CPU Unit	Model number	PLC model in Change PLC Dialog Box in CX-Programmer version 9.0 or higher
CJ Series	CJ2H CPU Unit	CJ2H-CPU6□-EIP CJ2H-CPU6□	CJ2H
	CJ2M CPU Unit	CJ2M-CPU3□ CJ2M-CPU1□	CJ2M

Troubleshooting Problems with Unit Versions on the CX-Programmer

Problem	Cause	Solution
 After the above message is displayed, a compiling error will be displayed on the *Compile* Tab Page in the Output Window.	An attempt was made to download a program containing instructions supported only by later unit versions or a CPU Unit to a previous unit version.	Check the program or change to a CPU Unit with a later unit version.
	An attempt was to download a PLC Setup containing settings supported only by later unit versions or a CPU Unit to a previous unit version.	Check the settings in the PLC Setup or change to a CPU Unit with a later unit version.
"????" is displayed in a program transferred from the PLC to the CX-Programmer.	An attempt was made to upload a program containing instructions supported only by higher versions of CX-Programmer to a lower version.	New instructions cannot be uploaded to lower versions of CX-Programmer. Use a higher version of CX-Programmer.

Related Manuals

Manuals related to a PLC built using a CJ-series CJ2 CPU Unit are listed in the following table. Use these manuals for reference.

Manual	Cat. No.	Model	Application	Description
CJ-series CJ2 CPU Unit Software User's Manual (this manual)	W473	CJ2H-CPU6□-EIP CJ2H-CPU6□ CJ2M-CPU□□	Software specifications for CJ2 CPU Units	Describes the following for CJ2 CPU Units: • CPU Unit operation • Internal memory • Programming • Settings • Functions built into the CPU Unit Also refer to the *Hardware User's Manual* (W472)
CJ-series CJ2 CPU Unit Hardware User's Manual	W472	CJ2H-CPU6□-EIP CJ2H-CPU6□ CJ2M-CPU□□	Hardware specifications for CJ2 CPU Units	Describes the following for CJ2 CPU Units: • Overview and features • Basic system configuration • Part nomenclature and functions • Mounting and setting procedure • Remedies for errors • Also refer to the *Software User's Manual* (W473).
EtherNet/IP Units Operation Manual	W465	CJ2H-CPU6□-EIP CJ2M-CPU3□ CS1W-EIP21 CJ1W-EIP21	Using the built-in EtherNet/IP port of the CJ2 CPU Unit	Describes the built-in EtherNet/IP port and EtherNet/IP Units. Describes basic settings, tag data links, FINS communications, and other functions.
CS/CJ/NSJ-series Instructions Reference Manual	W474	CJ2H-CPU6□-EIP CJ2H-CPU6□ CJ2M-CPU□□ CS1G/H-CPU□□H CS1G/H-CPU□□-V1 CJ1G/H-CPU□□H CJ1G-CPU□□ CJ1M-CPU□□ NSJ□-□□□□(B)-G5D NSJ□-□□□□(B)-M3D	Information on instructions	Describes each programming instruction in detail. Also refer to the *Software User's Manual* (W473) when you do programming.
CS/CJ/CP/NSJ-series Communications Command Reference Manual	W342	CJ2H-CPU6□-EIP CJ2H-CPU6□ CJ2M-CPU□□ CS1G/H-CPU□□H CS1G/H-CPU□□-V1 CS1D-CPU□□H CS1D-CPU□□S CS1W-SCU□□-V1 CS1W-SCB□□-V1 CJ1H-CPU□□H-R CJ1G/H-CPU□□H CJ1G-CPU□□P CJ1M-CPU□□ CJ1G-CPU□□ CJ1W-SCU□□-V1 CP1H-X□□□□-□ CP1H-XA□□□□-□ CP1H-Y□□□□-□ CP1L-M/L□□□-□ CP1E-E□□D□-□ CP1E-N□□D□-□ NSJ□-□□□□(B)-G5D NSJ□-□□□□(B)-M3D	Information on communications for CS/CJ/CP-series CPU Units and NSJ-series Controllers	Describes C-mode commands and FINS commands Refer to this manual for a detailed description of commands for communications with the CPU Unit using C mode commands or FINS commands. **Note** This manual describes the communications commands that are addressed to CPU Units. The communications path that is used is not relevant and can include any of the following: serial ports on CPU Units, communications ports on Serial Communications Units/Boards, and Communications Units. For communications commands addressed to Special I/O Units or CPU Bus Units, refer to the operation manual for the related Unit.
CX-One Setup Manual	W463	CXONE-AL□□C-V□/ AL□□D-V□	Installing software from the CX-One	Provides an overview of the CX-One FA Integrated Tool Package and describes the installation procedure.

Manual	Cat. No.	Model	Application	Description
CX-Programmer Operation Manual	W446	WS02-CX□□-V□	Support Software for Windows computers CX-Programmer operating procedure	Describes operating procedures for the CX-Programmer. Also refer to the *Software User's Manual* (W473) and *CS/CJ/NSJ-series Instructions Reference Manual* (W474) when you do programming.
CX-Programmer Operation Manual Functions Blocks/Structured Text	W447			
CX-Programmer Operation Manual SFC Programming	W469			
CS/CJ/CP/NSJ-series CX-Simulator Operation Manual	W366	WS02-SIMC1-E	Operating procedures for CX-Simulator Simulation Support Software for Windows computers Using simulation in the CX-Programmer with CX-Programmer version 6.1 or higher	Describes the operating procedures for the CX-Simulator. When you do simulation, also refer to the *CX-Programmer Operation Manual* (W446), *Software User's Manual* (W473), and *CS/CJ/NSJ-series Instructions Reference Manual* (W474).
CS/CJ/CP/NSJ-series CX-Integrator Network Configuration Software Operation Manual	W464	CXONE-AL□□C-V□/ CXONE-AL□□D-V□	Network setup and monitoring	Describes the operating procedures for the CX-Integrator.

1

Overview

This section provides an overview of the CJ2 CPU Units.

1-1 Overview of CJ2 CPU Units 1-2
1-1-1 Overview 1-2
1-1-2 CJ2 CPU Unit Features 1-4
1-2 Basic Operating Procedure 1-11

1-1 Overview of CJ2 CPU Units

1-1-1 Overview

The SYSMAC CJ2-series CPU Units are multi-functional CPU Units that provide the following features.

● Fast, with Large Memory Capacity

Basic performance is faster and memory capacity has been increased to provide ample capability for machine control.

● Built-in EtherNet/IP Port (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only)

The CJ2 CPU Units support the EtherNet/IP open network as a standard feature. Universal Ethernet communications, such as data links between PLCs, message communications between PLCs, and FTP transfers, are all possible from a peripheral device connection.

● General-purpose Networks for Support Software Interface

Support Software and devices can be easily connected using commercially available cable to general-purpose networks via USB and EtherNet/IP ports. (The EtherNet/IP port is provided only on the CJ2H-CPU6□-EIP and CJ2M-CPU3□.)

● Tag Access (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only)

CJ2 CPU Units have a tag name server to manage tag names and I/O addresses. This enables access from external devices using tag names, without needing to know the I/O addresses.

● Synchronous Unit Operation (CJ2H CPU Units with Unit Version 1.1 or Later)

The timing of processing performed by CPU Bus Units and Special I/O Units can be synchronized. This minimizes fluctuations in timing from input and processing to outputs, making it easier to ensure application performance.

● Easier Programming

CJ2 CPU Units offer a highly readable programming environment, including features such as addressing DM and EM Area bits, setting address offsets, and using array variables.

● Improved Debugging

Online editing and data tracing have been improved, greatly increasing the efficiency of debugging.

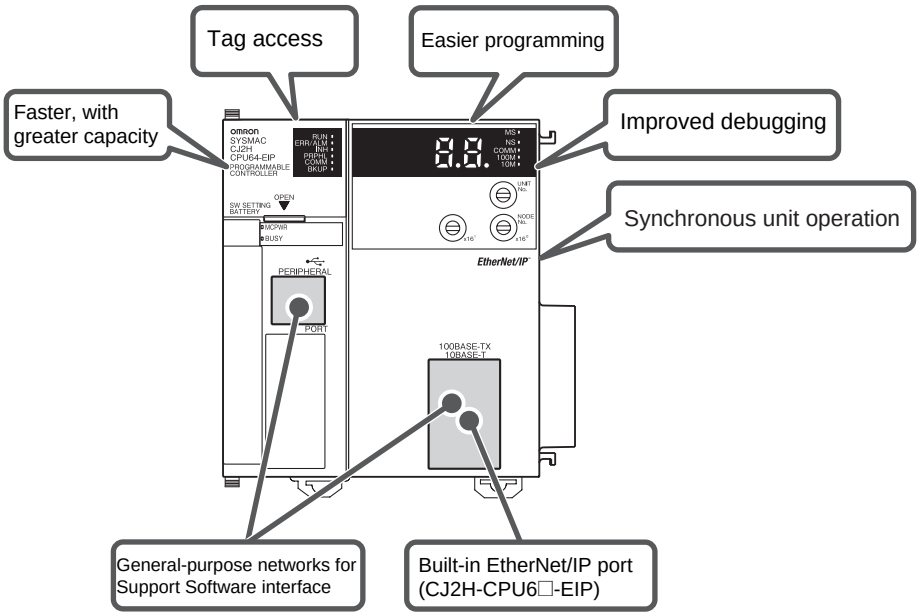
● Increased Pulse I/O Capacity (CJ2M CPU Unit with Unit Version 2.0 or Later)

With the CJ2M CPU Units, an optional Pulse I/O Block can be mounted to enable pulse I/O for up to four axes.

● More Serial Communications Ports (CJ2M-CPU3□ Only)

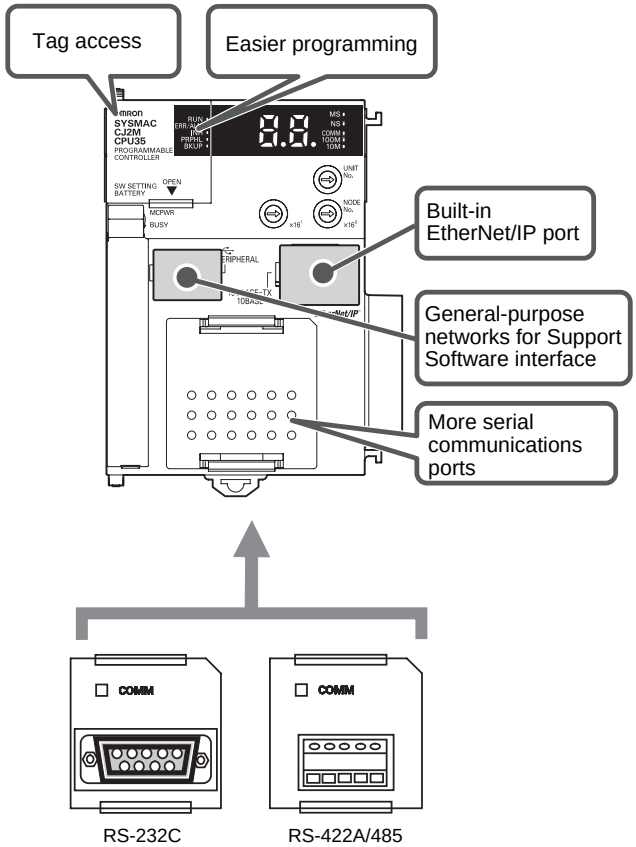
With the standard CJ2M CPU Unit (CJ2M-CPU3□), you can add an RS-232C or RS-422A/485 Option Board to the standard-feature EtherNet/IP port to increase the number of serial communications ports.

CJ2H CPU Units

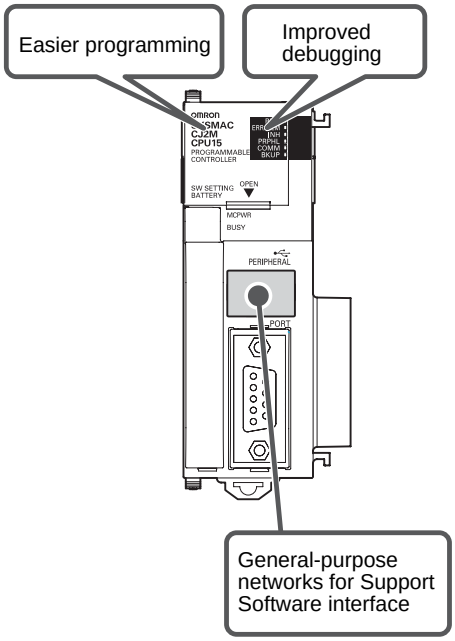


CJ2M CPU Units

Standard CPU Units (CJ2M-CPU3□)



Simple CPU Units (CJ2M-CPU1□)



1-1-2 CJ2 CPU Unit Features

Fast, with Large Memory Capacity

● High-speed Processing

High-speed processing is possible for basic instructions (0.016 μs min. for CJ2H and 0.04 μs min. for CJ2M), special instructions (0.05 μs min. for CJ2H and 0.06 μs min. for CJ2M), and immediate refreshing (0.99 μs min. for CJ2H and 1.26 μs min. for CJ2M).

● Large Program Capacity

The CJ2 CPU Units provide a large program capacity of up to 400 Ksteps.

● Special Function Block Definition Area (CJ2M Only)

With a CJ2M CPU Unit, a special area called the FB Program Area is provided to store function block definitions. (The FB Program Area holds 20K steps.) This allows you to use function blocks to make program components and structures from previous programs while reducing the usage of the User Program Area. And any function block definitions that would exceed the FB Program Area are stored in the User Program Area.

● Large Data Memory Capacity

The CJ2 CPU Units provide a large EM Area capacity of up to 800 Kwords (25 banks).

● Up to 128 Cyclic Tasks

The user program can be divided into up to 128 tasks. Using smaller task programs makes it easier to structure programs and contributes to shorter cycle times.

● Better Execution Performance for Interrupt Tasks (CJ2H CPU Unit with Unit Version 1.1 or Later)

With CJ2H CPU Units with unit version 1.1, overhead time for interrupt tasks is approximately 20% less than for unit version 1.0 even for normal usage. Also, by using High-speed interrupt function, it is possible to improve execution performance as shown below with certain restrictions.

- Greatly reduce overhead time for interrupt tasks (interrupt task startup time + return time to cyclic tasks).

  Example: For I/O interrupt tasks, the time for normal operation is 37 μs but the time is 25 μs if High-speed interrupt function is used.

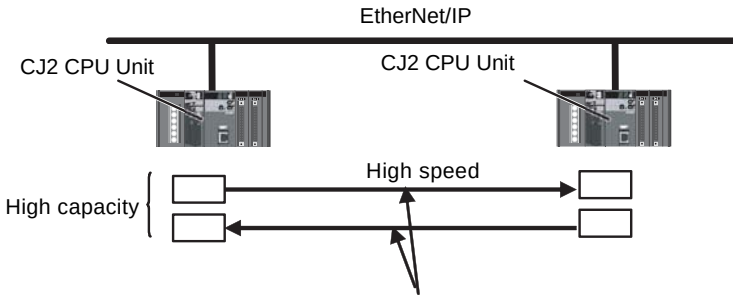
- Minimum Interval Setting of 0.1 ms for Scheduled Interrupt Tasks
  (For unit version 1.0, the minimum interval is 0.2 ms.)

Built-in EtherNet/IP Port (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only)

The CJ2 CPU Units provide as standard equipment a built-in EtherNet/IP port that supports the same FINS Ethernet functions as an Ethernet Unit (including features such as a FTP server and time coordination) as well as the EtherNet/IP open network.

● High-speed, High-capacity Data Links with User-set Refresh Cycles

The CJ2 CPU Units provide high-speed, high-capacity data links, and refresh cycles can be set individually for each data link area.



The refresh cycle can be set individually for each area.

In addition, using the EtherNet/IP DataLink Tool makes it possible to set the data links using an interface similar to that of the Controller Link automatic data link setup.

General-purpose Networks for Support Software Interface

● Peripheral USB and EtherNet/IP Ports

A commercially available USB cable can be used to connect to the PLC directly from a USB port on a personal computer. In addition, with the CJ2H-CPU6□-EIP or CJ2M-CPU3□, a PLC on the EtherNet/IP network can be accessed via USB.

● Prevent Connecting to the Wrong PLC by Using PLC Names from Support Software

A user-set PLC name can be recorded in a CJ2 CPU Unit. When using Support Software to connect online to a PLC, verification of the PLC name prevents incorrect connections from the Support Software.

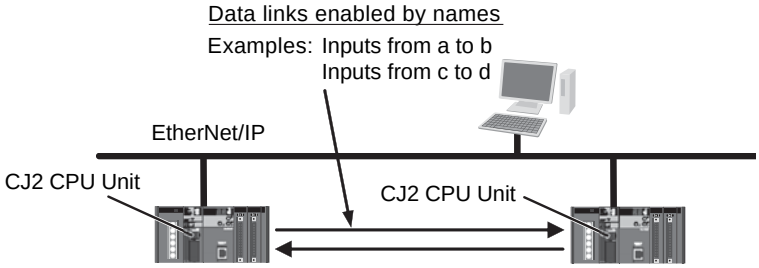
Tag Access (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only)

● Network Symbols (Tags) for Flexible Support of Program Changes

The CJ2 CPU Units support network symbols (tags). They have an internal tag name server that enables them to store tag names and addresses in advance in symbol tables in the CPU Units. Tags enable the following features.

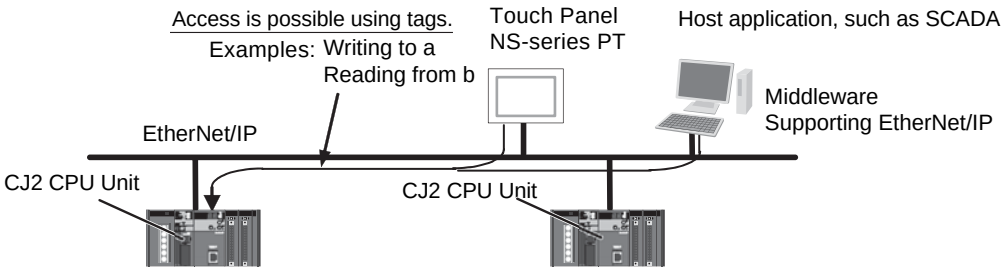
Data links can be set using tags.

With data links on an EtherNet/IP network, send and receive areas can be specified using symbols rather than addresses. This enables flexible support for design changes by allowing the data link areas set by tags to remain unchanged, while simply changing the symbol tables that contain the tag names and addresses.



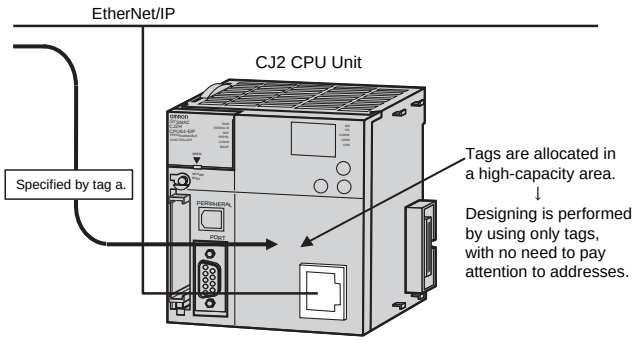
Access from host devices is enabled using tags.

Tags can be used for access from host devices, such as touch panel NS-series Programmable Terminals. This makes the creation of host screens and programs at devices such as PTs much more convenient.



Automatic tag allocation makes it unnecessary to know the addresses.

Automatic allocation of tags in the high-capacity EM Area, using automatic address allocation in CX-Programmer symbol tables, enables data link design and access from host devices without having to pay attention to addresses.

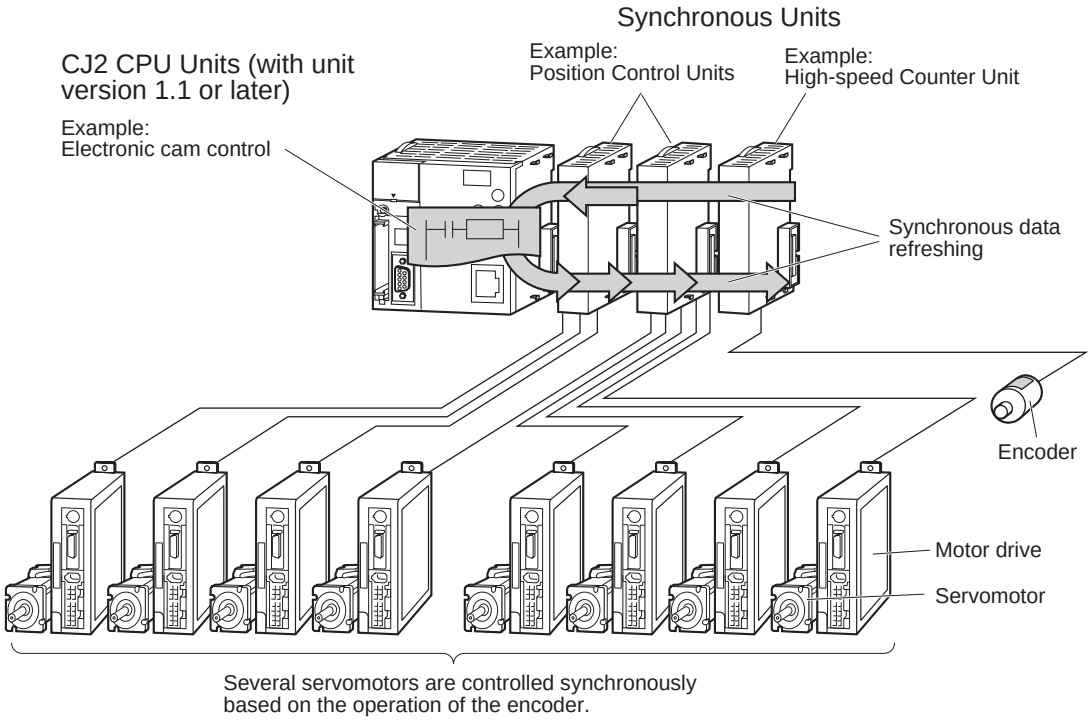


Synchronous Unit Operation (CJ2H CPU Units with Unit Version 1.1 or Later)

● A CJ2 CPU Unit can synchronize the start of the processing performed by CPU Bus Units and Special I/O Units to a specified cycle.

Synchronous data refreshing can be used between the CPU Unit and Synchronous Units,* and the refreshed data can be used in operations in a synchronous interrupt task. This enables multiaxis control with minimal fluctuations in the timing from inputs and processing to outputs. This provides support for motion applications that require precise timing, such as electronic cam control.

* Synchronous Units are CPU Bus Units and Special I/O Units that support the synchronous unit operation function. Only the CJ1W-NC214, CJ1W-NC234, CJ1W-NC414, and CJ1W-NC434 Position Control Units support synchronous unit operation as of December 2009.



Easier Programming

● Address offsets can be specified.

When an address is specified for an instruction operand, an offset can be specified in brackets after the address to offset it. For example, by setting a word address in brackets to specify the offset, the address can be dynamically specified according to the contents of that word.

Example
W0.00[D0]: W0.00 is the starting address and the contents of D0 is the offset. If D0 is &3, then W0.03 is specified.

● Symbols can be specified for array variable subscripts.

By specifying symbols for array variable subscripts, elements can be dynamically specified according to the values of the symbols.

Example
a[b]: The value of symbol b specifies the element for array variable a[ ].

● Create and Use Data Structures

With CX-Programmer version 9.0 or higher, data structures can be created and used. This enables easily using I/O memory in the CPU Unit as a database or for library data.

● Bit addresses can be used in the DM Area and EM Area.

Previously the DM Area and the EM Area could be addressed only by words, and bit addresses could not be specified. The work area for bits can now be expanded by enabling bit addresses in the DM and EM Areas.

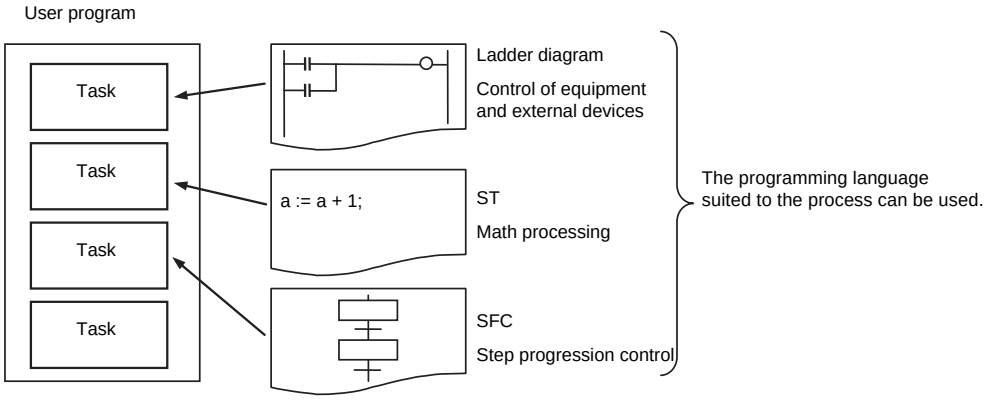
For example, D10.00 specifies bit 00 of D10.

● The format for timer/counter PV refreshing can be selected individually for each instruction.

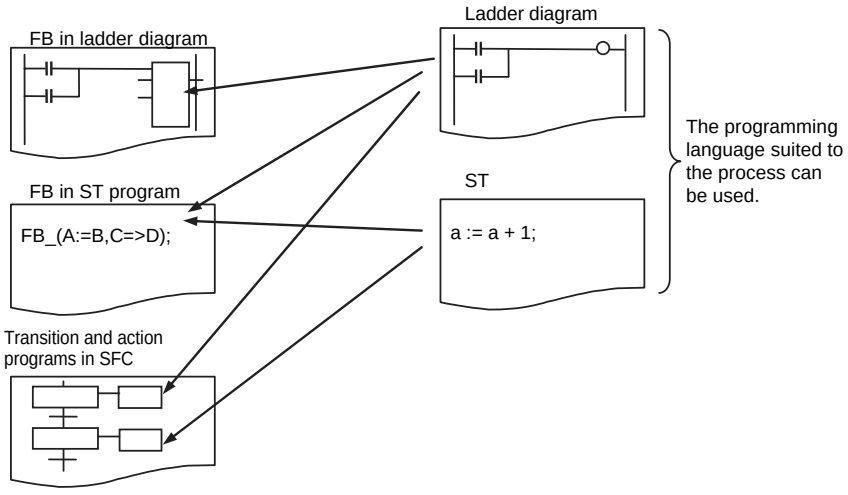
Either BCD or binary can be selected individually for each instruction as the format for timer/counter PV refreshing. For example, the TIM (BCD) and TIMX (binary) instructions can be used together.

● The optimum languages can be combined for structured programming.

Ladder diagrams, ST, and SFC languages can be freely combined in the user program. Being able to use the optimum languages for a particular process helps reduce the time involved in programming.



In addition, ladder diagrams and ST language can be freely used for function blocks (FBs) in ladder diagrams and ST programs, and for transition and action programs in SFC. This feature makes it possible to optimize structured programming.



Improved Debugging

● Reduced Effect on Cycle Time from Online Editing

The additional cycle time due to online editing has been reduced to approx. 1 ms, minimizing the effects on equipment operation during debugging.

● Fast, High-capacity Data Tracing

Up to 32 Kwords of data can be traced (8 time more than previously), with ample trigger conditions, and data can be traced continuously for long periods of time.

● Easy Setup Function for Data Tracing with CX-Programmer Version 9.0 or Higher

With CX-Programmer version 9.0 or higher, data to be traced can be easily selected from lists. When using the Pulse I/O Block with a CJ2M CPU Unit, the current pulse output frequency can be calculated every 500 μs and the data can be traced.

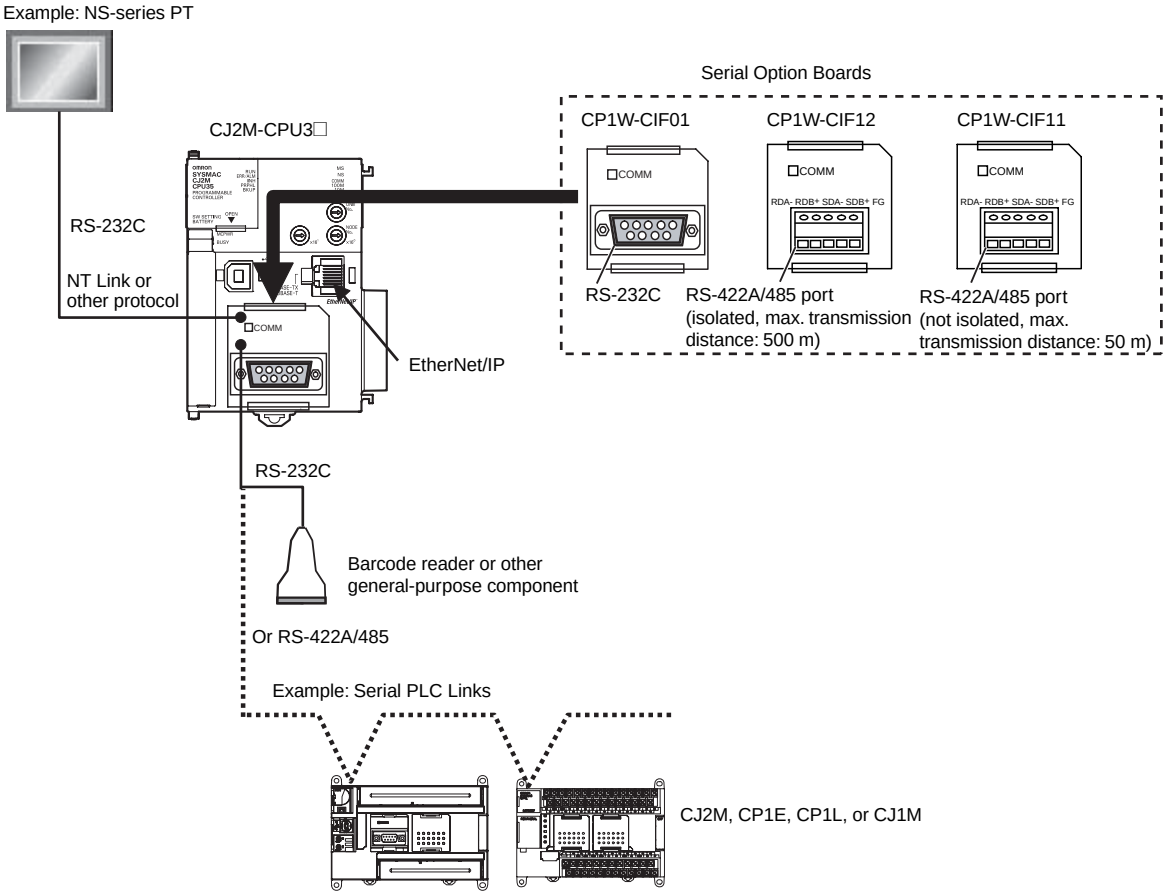
● Force-set/Reset Bits in Specified EM Area Banks

With CX-Programmer version 8.3 or higher, a parameter can be set to enable force-setting/resetting bits in specified EM Area banks. (This function is supported only by CJ2H CPU Units with unit version 1.2 or later and CJ2M CPU Units.)

Add a Serial Option Board (CJ2M-CPU3□ Only)

With the CJ2M-CPU3□, one Serial Communications Option Board with one RS-232C port or one RS-422A/485 port can be added.

With the serial port, it is easy to connect to general components, such as barcode readers, and other components such as PTs, other CJ/CP-series PLCs, and Inverters.



1-2 Basic Operating Procedure

Use Pulse I/O and Interrupt Inputs (CJ2M CPU Units Only)

In general, use the following procedure.

1. Setting Devices and Hardware

Mount the Power Supply Unit, the CPU Unit, the other Units, and the End Covers. Set the DIP switch and rotary switches as required.

Refer to *Section 3 Nomenclature and Functions* and *Section 5 Installation* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).

2. Wiring

Wire the power supply, I/O, and communications. Refer to the following manuals.

Refer to *Section 5 Installation* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).
Refer to *Section 11 Programming Devices and Communications* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).

3. Connecting Online to the PLC

Connect the personal computer online to the PLC.

Refer to *Section 4 Support Software* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472)

4. I/O Allocations

Using the CX-Programmer, allocate I/O memory in the CPU Unit to the mounted Units.

Refer to *Section 8 I/O Allocations and Unit Settings* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473)

5. Software Setup

Make the PLC software settings.

- CPU Unit initialization: Refer to *Section 4 CPU Unit Initialization* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).
- DM Area Settings for Special I/O Units and CPU Bus Units: Refer to *Section 8 I/O Allocations and Unit Settings* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).
- PLC Setup: Refer to *Section 9 PLC Setup* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).

6. Creating the Program

Create the program using the CX-Programmer.

Refer to *Section 5 Understanding Programming* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).

7. Checking Operation

Check the I/O wiring and the AR Area settings, and perform trial operation. The CX-Programmer can be used for monitoring and debugging.

Refer to *10-7 Debugging* in the *CJ2 CPU Unit Software User's Manual* (Cat. No. W473).

8. Basic Program Operation

Set the operating mode to RUN mode to start operation.

2

Internal Memory in the CPU Unit

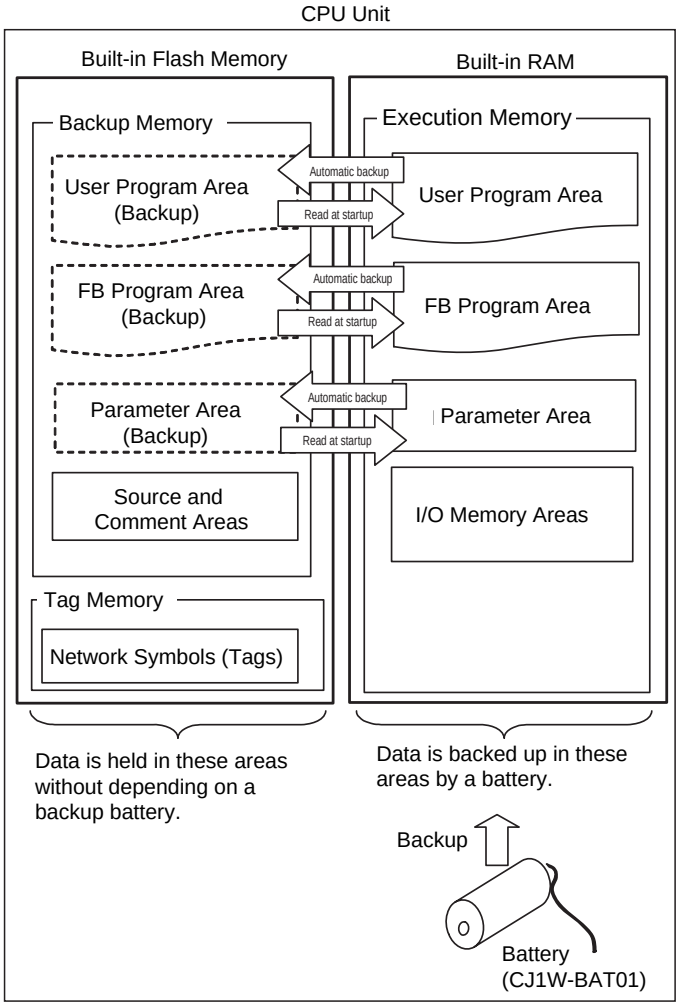
This section describes the memory areas contained in the CPU Unit.

2-1 Overview 2-2
2-1-1 Memory Configuration 2-2
2-1-2 Memory Areas and Stored Data 2-3
2-1-3 Transferring Data from a Programming Device to the CPU Unit 2-4

2-1 Overview

2-1-1 Memory Configuration

As shown in the following diagram, the internal memory in the CPU Unit consists of built-in flash memory and built-in RAM. The built-in RAM is used as execution memory and the built-in flash memory is used as backup memory.



Built-in Flash Memory

The built-in flash memory backs up the user program and parameters, and stores the program source, comment, and network symbols. Data is held in this area even without depending on a battery. I/O memory is not backed up.

Built-in RAM

The built-in RAM is the execution memory for the CPU Unit. The user program, parameters, and I/O memory are stored in the built-in RAM, and the built-in RAM is backed up by battery.
If the battery does not work (e.g., if the battery voltage is low or no battery is installed), the I/O memory data is lost. The user program and parameters are backed up to the built-in flash memory, so they are not lost.

Precautions for Correct Use

The following will occur if the battery is low or when no battery is installed.

- Data in the I/O memory areas will be lost or values will become unstable, including values in the DM, EM, and HR Areas, which are retained by the battery when power is OFF.
- The clock will stop, and all clock-related data will become unstable.
- Error logs will not be retained.
- The Output OFF Bit will become unstable.

2-1-2 Memory Areas and Stored Data

The following table lists the CPU Unit memory areas and the data stored in each area.

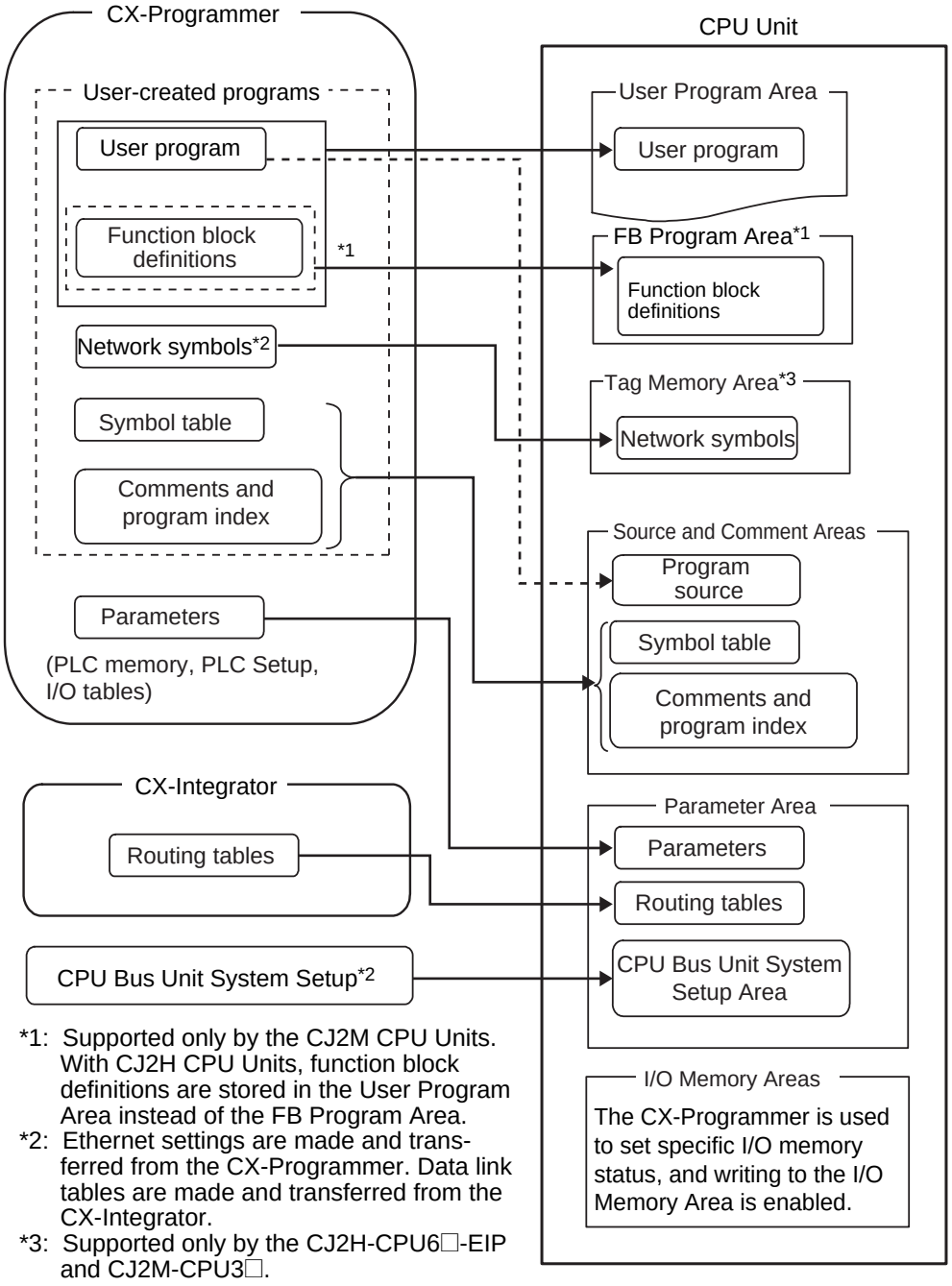
Memory area and stored data	Details	Built-in flash memory	Built-in RAM
User Program Area	The User Program Area stores the object code for executing the user program that was created using the CX-Programmer.	Stored.	Stored.
FB Program Area*1	The FB Program Area stores the function block definitions created using the CX-Programmer.	Stored.	Stored.
Parameter Area	The Parameter Area stores the initial settings for the PLC.	Stored.	Stored.
PLC Name	The name of the CPU Unit is stored and can be read and verified by the CX-Programmer to prevent the CX-Programmer from connecting to the wrong PLC.		
PLC Setup	Various initial settings are made in the PLC Setup using software switches. Refer to *Section 9 PLC Setup*.		
I/O Tables	I/O tables provide information on the mounting status of Units specified by the user. Refer to *Section 8 I/O Allocations and Unit Settings*.		
Routing Tables	Routing tables are network parameters for FINS communications. They are specified using the CX-Integrator. Refer to the *CX-Integrator Operation Manual* (Cat. No. W464).		
CPU Bus Unit Setup	The CPU Bus Unit Setup stores the initial settings for specific CPU Bus Units. It includes settings such as Ethernet settings for Ethernet Units and data link parameters for Controller Link Units.		
I/O Memory Areas	The I/O Memory Areas are used for reading and writing from the user program. It is partitioned into the following regions according to purpose. A region where data is cleared when power to the CPU Unit is reset, and a region where data is retained. A region where data is exchanged with other Units, and a region that is used internally.	---	Stored
Source and Comment Areas	The Source and Comment Areas are used for storing the program source code and comments created using the CX-Programmer.	Stored	---
Source Code	The source code for programs (in tasks and function blocks, using ladder, ST, and SFC languages).		
Symbol Table	The symbol table contains symbols created using the CX-Programmer (symbol names, addresses, and I/O comments).		
Comments	Comments are created using the CX-Programmer and include annotations and row comments.		
Program Index	The program index provides information on program sections created using the CX-Programmer, as well as program comments.		
Network Symbols (Tags)*2	Data for network symbols in the global symbol table.	Stored	---

*1 Supported only by the CJ2M CPU Units. With CJ2H CPU Units, function block definitions are stored in the User Program Area instead of the FB Program Area.

*2 Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.

2-1-3 Transferring Data from a Programming Device to the CPU Unit

Data that has been created using the CX-Programmer or the CX-Integrator is transferred to the internal memory in the CPU Unit as shown in the following diagram.



*1: Supported only by the CJ2M CPU Units. With CJ2H CPU Units, function block definitions are stored in the User Program Area instead of the FB Program Area.
*2: Ethernet settings are made and transferred from the CX-Programmer. Data link tables are made and transferred from the CX-Integrator.
*3: Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.

3

CPU Unit Operation

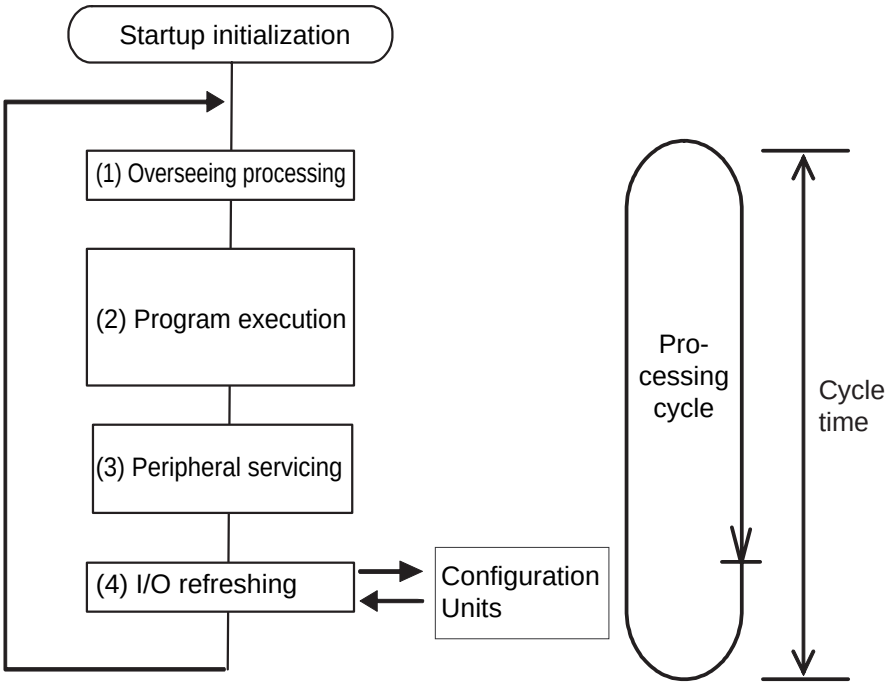
This section describes the internal operation of the CPU Unit and the operating modes that can be set for the CPU Unit.

3-1 CPU Unit Internal Operation 3-2
3-1-1 Overview 3-2
3-1-2 Cycle Time 3-4
3-1-3 Processing at Power Interruptions 3-7
3-2 CPU Unit Operating Modes 3-8
3-2-1 Operating Modes 3-8
3-2-2 Checking the Operating Mode 3-9
3-2-3 Changing the Operating Mode 3-10
3-2-4 Operating Mode Details 3-14

3-1 CPU Unit Internal Operation

3-1-1 Overview

The CPU Unit writes data to the internal I/O memory areas while it cyclically executes user programs. Data is exchanged externally when I/O is refreshed and peripherals are serviced. The following figure shows the internal operation of the CPU Unit.



Startup Initialization

The following processes will be performed once each time the power is turned ON to the PLC.

- Detecting connected Units (I/O allocation)
- Comparing the registered I/O tables and the connected Units
- Clearing the non-holding areas in I/O memory according to the status of the IOM Hold Bit
- Clearing forced status according to the status of the Forced Status Hold Bit
- Autobooting using the autotransfer files in the Memory Card if one is inserted
- Performing self-diagnosis (user memory check)
- Restoring the user program
- Updating the PLC Setup

Processing Cycle

The CPU Unit will repeatedly perform four processes (overseeing, program execution, I/O refreshing, and peripheral servicing) after startup initialization. The time it takes to complete one cycle is called the cycle time.

- Overseeing
  This process prepares to execute the user program. It checks the hardware and software required for processing. The time required for overseeing is called the overhead time.
- Program Execution
  This process executes the instructions in the user program. The instructions in the user program are executed when the operating mode is set to RUN or MONITOR mode. I/O processing for the instructions is performed on bits and words in the I/O memory areas in the CPU Unit.
- Peripheral Servicing
  This process occurs when messages are required from Programming Devices or Units. Data is written from the CX-Programmer or performed for instructions for Network Communications Units.
- I/O Refreshing
  This process exchanges data with the Units by using the I/O memory areas. This process is always performed after program execution. I/O refreshing is performed within the current cycle without interruption (i.e., without time slicing).

3-1-2 Cycle Time

The cycle time is the total time for overseeing, program execution, peripheral servicing, and I/O refreshing. The cycle time normally fluctuates each cycle depending on the processing that is performed. Each of these processes is performed within the cycle time. Refer to *Section 12 CPU Unit Cycle Time* for information on how to calculate the cycle time.



Additional Information

A constant cycle time can be created by setting a minimum cycle time in the PLC Setup. The minimum cycle time must be longer than any normal cycle time. For information on how to set a minimum cycle time, refer to *10-2-1 Minimum Cycle Time*.

Overseeing

The overhead time occurs for overseeing each cycle. The following items are checked.

- Battery error
- Memory Card
- DIP switch
- I/O bus
- User program memory

Program Execution

This is the processing time for instructions in the user program.

- The processing time includes instructions in cyclic tasks that are in READY status and instructions in interrupt tasks for which the interrupt conditions are met. It does not include instructions that are jumped in the programs.
- The processing time depends on the number of instructions and the specific instructions that are used. It also depends on the operands that are specified (e.g., on the amount of data to be processed).
- In RUN or MONITOR mode, internal processing time is required for instructions with OFF execution conditions. In the following cases, internal processing is not performed and processing time is not required.
  - A cyclic task in WAIT status
  - Instructions that are jumped for a JMP(004), CJP(510), or CJPN(511) instruction.



Additional Information

Consider the following programming if it is necessary to shorten the cycle time.

- Divide programs into cyclic tasks and place tasks into WAIT status when they do not need to be executed.
- Use the JMP(004), CJP(510), and CJPN(511) instructions to jump instructions when they do not need to be executed.

Peripheral Servicing

Peripheral servicing involves servicing non-scheduled events for external devices. This includes both events from external devices and service requests to external devices.

Most peripheral servicing for CJ-series PLCs involves FINS commands. The amount of time specified in the system is allocated to each type of servicing and executed every cycle. If all servicing cannot be completed within the allocated time, the remaining servicing is performed the next cycle. If peripheral servicing is delayed too much, increase the fixed servicing time in the PLC Setup to a value larger than 10% using the CX-Programmer.

● Peripheral Servicing Time

With the default peripheral servicing time setting, 10% of the preceding cycle time is assigned to the total peripheral service. Therefore, the peripheral servicing time depends on the preceding cycle time.

Set a time longer than 10% of the cycle time using Fixed Servicing Time in the PLC Setup from the CX-Programmer when peripheral servicing is delayed because too many cycles is required to complete it. Keep in mind that increasing the servicing time will also increase the cycle time.

● Types of Peripheral Servicing

Units	Servicing
Event servicing for Special I/O Units	• Non-scheduled servicing for FINS commands from Special I/O Units and CPU Bus Units (e.g., requests to start external interrupt tasks) • Non-scheduled servicing for FINS commands from the CPU Unit to the above Units.
Event servicing for CPU Bus Units	
Peripheral USB port service	• Non-scheduled servicing for FINS or Host Link commands received via the peripheral USB or serial ports from Programming Devices, PTs, or host computers (e.g., requests to transfer programming, monitoring, forced-set/reset operations, or online editing) • Non-scheduled servicing from the CPU Unit transmitted from the peripheral USB or serial port (non-solicited communications)
Serial port service	
Communications port servicing	• Servicing to execute network communications, serial communications, or file memory access for the SEND(090)/SEND2(491), RECV(098)/RECV2(492), CMND(490)/CMND2(493), or PMCR(260)/PMCR2(264) instructions using communications ports (internal logic ports). • Servicing to execute background execution using communications ports.
File access servicing	File read/write operations for Memory Cards or EM file memory.
Online editing servicing	Servicing to backup contents edited with online editing in the built-in flash memory when online editing is being processed.
Backup servicing	Servicing to backup changed contents in the built-in flash memory when there is a change in programming or parameters.

Additional Information

Servicing time is allocated to Special I/O Units, CPU Bus Units, the peripheral USB port, serial ports, file access, and communications ports.
If servicing is separated over many cycles, delaying completion of the servicing, set the same allocated time (same time for all services) rather than a percentage under execute time settings in the PLC Setup.

I/O Refreshing

The I/O Unit processing time depends on the type and number of Configuration Units that are used in the PLC. The processing time for each Unit is constant.

It is possible to shorten the processing time by stopping cyclic I/O refreshing for Special I/O Units. To stop I/O refreshing for Special I/O Units, set the parameter on the SIOU Refresh Tab Page in the CX-Programmer. The following table lists the refresh processing for the PLC Units.

Units			Maximum data exchange	Data exchange area
Basic I/O Units			Depends on the Unit.	I/O Bit Area
Special I/O Units	Words allocated in CIO Area		10 words/Unit (Depends on the Unit.)	Special I/O Unit Area
	Unit-specific data	CompoBus/S Master Unit	Depends on the Unit.	Words set for remote I/O communications
CPU Bus Units	Words allocated in CIO Area		25 words/Unit	CPU Bus Unit Area
	Words allocated in DM Area		100 words/Unit	CPU Bus Unit words
	Unit-specific data (Refer to the right.)	Built-in EtherNet/IP port* EtherNet/IP Unit	Depends on the EtherNet/IP functions that are used.	Tag Data Link Area
		Controller Link Unit and SYSMAC LINK Unit	Depends on the Unit.	Words set for data links (for either fixed or user-set allocations)
		DeviceNet Unit	Depends on the Unit.	Words set for remote I/O communications (for either fixed or user-set allocations)
		Serial Communications Unit	Depends on the protocol macros.	Communications data set for protocol macros
		Ethernet Unit	Depends on the Unit.	Communications data for socket services initiated by specific control bit operations.

* Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□. The EtherNet/IP port built into the CJ2H-CPU6□-EIP provides the same performance and functions as a CJ1W-EIP21 EtherNet/IP Unit. The EtherNet/IP port built into the CJ2M-CPU3□ provides very different performance. Refer to the *EtherNet/IP Unit Operation Manual* (Cat. No. W465) for details.



Additional Information

It is possible to exchange data with the Units when instructions are executed rather than during the normal I/O refresh period. This is called immediate refreshing. Immediate refreshing is possible by attaching an exclamation mark (!) to some instructions, or by using the IORF(097), FIORF(225), and DLNK(226) instructions.

The following instructions can be used to increase the speed of data exchange with certain Special I/O Units or CPU Bus Units by using direct processing. Refer to the *Instructions Reference Manual* (Cat. No. W474) for details.

- ANALOG INPUT DIRECT CONVERSION (AIDC(216)) (for CJ1W-AD042 High-speed Analog Input Unit)*1
- ANALOG OUTPUT DIRECT CONVERSION (AODC(217)) (for CJ1W-DA042V High-speed Analog Output Unit)*1
- DIRECT TRANSMIT VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (DTXDU(262)) (CJ1W-SCU□2 only)*1
- DIRECT RECEIVE VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (DRXDU(261)) (CJ1W-SCU□2 only)*1
- PCU HIGH-SPEED POSITIONING (NCDMV(218)) (CJ1W-NC□□4 and CJ1W-NC□ only)*2
- PCU POSITIONING TRIGGER (NCDTR(219)) (CJ1W-NC□81 only)*2

*1 These instructions are supported only by the CJ2H-CPU6□(-EIP) (unit version 1.1 or later) and CJ2M CPU Units.

*2 Supported only by the CJ2H-CPU6□(-EIP) with unit version 1.3 or later and CJ2M CPU Units.

3-1-3 Processing at Power Interruptions

If power is interrupted and the interruption is confirmed while the CPU Unit is operating, the instruction currently being executed will be completed and the following power interruption processing will be performed.

Processing at power interruption will be performed according to the settings for power OFF interruption tasks.*1

- If the power OFF interrupt task has been enabled, the task will be executed and then the CPU Unit will be reset immediately.
- If the power OFF interrupt task has not been enabled, the CPU Unit will be reset*2 immediately.

*1 The power OFF interrupt task is executed immediately before the CPU Unit is reset due to power interruption.

*2 It is possible to perform power interruption processing after a specified range of instructions is executed when the power is interrupted during operation.

Operation will always continue for momentary power failures of less than 10 ms for an AC power supply and less than 2 ms for a DC power supply. It is possible to lengthen the time from which a power interruption is detected until it is confirmed as a power interruption when the power supply conditions are poor.

For information on processing at power interruption, refer to *A-5 Operation for Power Interruptions*.

3-2 CPU Unit Operating Modes

3-2-1 Operating Modes

The operating mode can be set to control the operating conditions of the CPU Unit and control whether settings can be made in the CPU Unit. There are three operating modes.

- **RUN mode:**

  RUN mode is used for actual operation of the system and provides the fastest operation.
  The programs are executed.
  Bits cannot be force-set/reset, values in I/O memory cannot be changed, and online editing is not possible.

- **MONITOR mode:**

  MONITOR mode is for trial operation and adjustment.
  The programs are executed.
  Bits can be force-set/reset, values in I/O memory can be changed, and online editing is possible.

- **PROGRAM mode:**

  PROGRAM mode is for transferring programs and the PLC Setup and creating the I/O tables.

  The programs are not executed.

Change the operating mode by using the CX-Programmer connected to the CPU Unit.

3-2-2 Checking the Operating Mode

Front-panel Indicator on the CPU Unit

The RUN indicator on the front of the CPU Unit indicates the operating mode as described below.

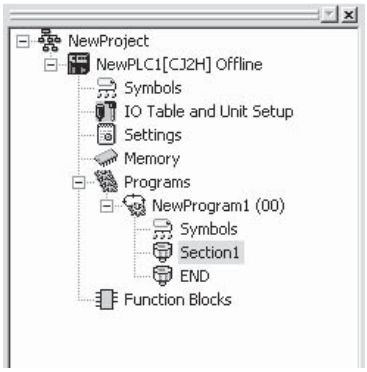
Operating mode	RUN indicator on CPU Unit	Remarks
PROGRAM mode	OFF	---
RUN or MONITOR mode	Lit green	Use the CX-Programmer to see if the mode is RUN or MONITOR mode.

CX-Programmer

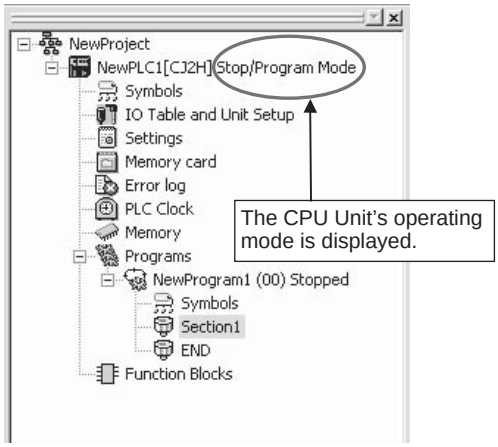
You can check the operating mode in the project tree or status bar of the CX-Programmer.

- **Project Tree**

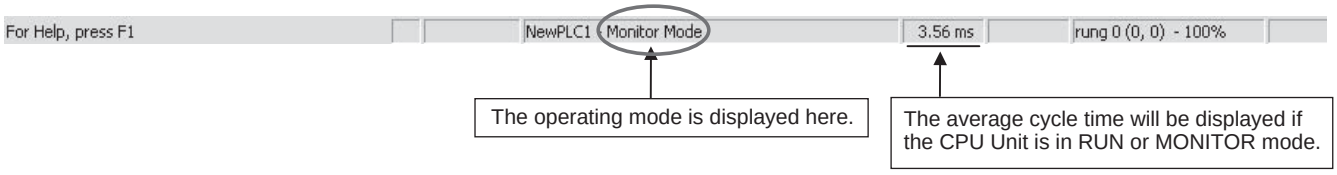
Offline



Online



- **Status Bar**



Additional Information

An external RUN output can be achieved by using the CJ1W-PA205R Power Supply. The RUN output (contact output) on the CJ1W-PA205R will turn ON in RUN or MONITOR mode unless there is a fatal error.

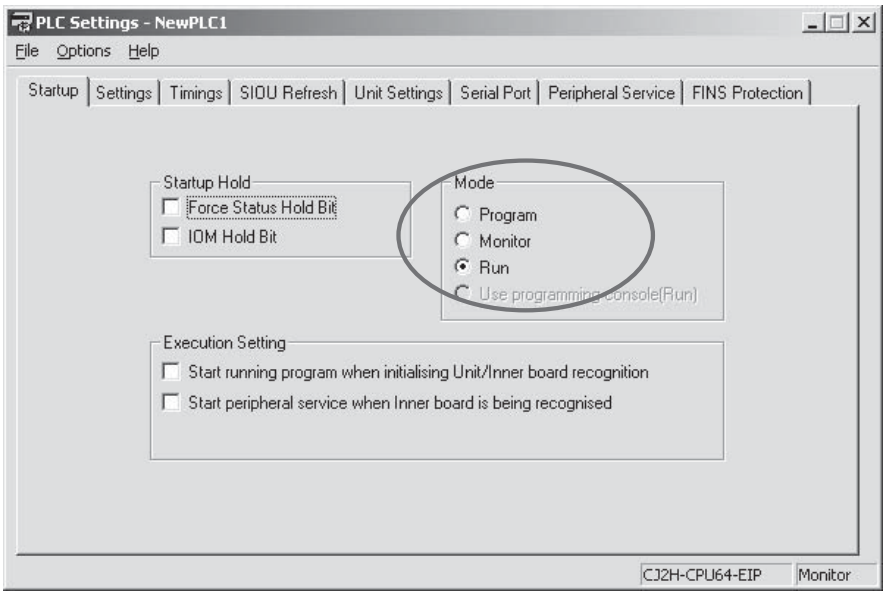
3-2-3 Changing the Operating Mode

Changing the Operating Mode

The operating mode can be changed from the CX-Programmer.

● Changing the Startup Mode

The default operating mode when the CPU Unit is turned ON is RUN mode. To change the startup mode to PROGRAM or MONITOR mode, set the desired mode in Startup Setting in PLC Setup from the CX-Programmer.



● Changing the Operating Mode after Startup

Use one of the following procedures.

- Select PROGRAM, MONITOR, or RUN from the Operating Mode Menu.
- Right-click the PLC in the project tree, and then select PROGRAM, MONITOR, or RUN from the Operating Mode Menu.

Memory Status after Mode Changes and Power Interruptions

Some parts of I/O memory, such as the CIO Area and Work area, will be cleared when the operating mode is changed between RUN or MONITOR mode and PROGRAM mode. These are called non-holding areas. To hold the contents of these areas when the operating mode is changed, turn ON the IOM Hold Bit (A500.12) in the Auxiliary Area

Mode change	Non-holding areas	Holding Areas
	• I/O Area • Data Link Area • CPU Bus Unit Area • Special I/O Unit Area • Work Area • Timer PV and Completion Flags • Index Registers • Data Registers • Task Flags (Auxiliary Area bits/words are held or not held depending on the address.)	• HR Area • DM Area • EM Area • Counter PV and Completion Flags (Auxiliary Area bits/words are held or not held depending on the address.)
RUN or MONITOR to PROGRAM	Cleared*1	Held
PROGRAM to RUN or MONITOR		
RUN to MONITOR or MONITOR to RUN	Held	Held
Power interruption reset*2	Cleared*3	Held

*1 Memory is cleared on if the IOM Hold Bit is OFF. If it is ON, data will be held as follows:

I/O Memory Hold Bit status (A500.12)	I/O Memory			Output bits allocated to Output Units		
	Mode changed between PROGRAM and RUN/MONITOR	Fatal error		Mode changed between PROGRAM and RUN/MONITOR	Fatal error	
		Fatal error other than FALS	FALS executed		Fatal error other than FALS	FALS executed
OFF	Cleared	Cleared	Held	OFF	OFF	OFF
ON	Held	Held	Held	Held	OFF: Values in the I/O memory in the CPU Unit will be held when a fatal error occurs, but the outputs from Basic Output Units will turn OFF.	

*2 The IOM Hold Bit (A500.12) will be turned OFF when a power interruption is reset. To retain the status, select the *IOM Hold Bit* Check Box in the Startup Hold Settings Tab Page of the PLC Setup using the CX-Programmer, and then transfer the PLC Setup.

*3 The Task Flags will be cleared regardless of the status of IOM Hold Bit (A500.12).

Output Status from Basic Output Units after Mode Change

When the operating mode is changed between RUN or MONITOR mode and PROGRAM mode with the default settings, the outputs from Basic Output Units will be refreshed according to the I/O memory status that exists when the mode is changed.

The output status for Basic Output Units will be as follows depending on whether IOM Hold Bit (A500.12) is ON or OFF.

- If A500.12 is OFF, the output bits will be cleared when the mode is changed to PROGRAM mode, and so the outputs from the Basic Output Units will turn OFF.
- If A500.12 is ON, the status of the output bits is held when the mode is changed to PROGRAM mode, and so the status of the outputs from the Basic Output Units will also be held.



Precautions for Correct Use

Precautions for setting the IOM Hold Bit (A500.12) to ON

When the mode is changed from RUN or MONITOR mode to PROGRAM mode, the I/O memory status of the outputs will not be cleared (i.e., turned OFF), and the status before changing to PROGRAM mode will be held. Therefore, when changing the mode again from PROGRAM mode to RUN or MONITOR mode, the I/O memory values before changing to PROGRAM mode will be output.

If operation stops due to a fatal error (including execution of an FALS(007) instruction), however, all outputs from the Basic Output Units will turn OFF even if the I/O memory status in the CPU Unit is held.

Output Status from Basic Output Units in PROGRAM Mode

Outputs from Basic Output Units will be refreshed with the I/O memory output bit status even in PROGRAM mode. Therefore, the status will be output from the Basic Output Units when the output bits in I/O memory are changed using the CX-Programmer or other Support Software.

⚠ WARNING

The CPU Unit refreshes I/O even when the program is stopped (i.e., even in PROGRAM mode). Confirm safety thoroughly in advance before changing the status of any part of memory allocated to Basic I/O Units, Special I/O Units, or CPU Bus Units. Any changes to the data allocated to any Unit may result in unexpected operation of the loads connected to the Unit. Any of the following operation may result in changes to memory status.

- Transferring I/O memory data to the CPU Unit from a Programming Device
- Changing present values in memory from a Programming Device
- Force-setting/-resetting bits from a Programming Device
- Transferring I/O memory files from a Memory Card or EM file memory to the CPU Unit
- Transferring I/O memory from a host computer or from another PLC on a network

To be sure that the outputs from the Basic Output Units remain OFF, turn ON the Output OFF Bit (A500.15). (See note.) The INH indicator on the front of the CPU Unit will be lit when all outputs are OFF (i.e., when the Output OFF Bit is ON).

Note The status of the Output OFF Bit (A500.15) is held when the operating mode is changed and the power is turned OFF and ON, i.e., the outputs will remain OFF.

3-2-4 Operating Mode Details

The following table shows the status during each operating mode.

Operating mode		PROGRAM	MONITOR	RUN
Purpose		Stopping operation and transferring programs	Testing operation and making adjustments	Main operation
Program status		Stopped	Executed	Executed
RUN indicator		OFF	ON	ON
Operation using CX-Programmer	Creating I/O tables	Possible	Not possible	
	Transferring PLC Setup			
	Downloading programs			
	Checking programs			
	Online editing	Possible		Not possible
	Checking wiring by force-setting/resetting bits			
	Changing I/O memory PVs			
	Changing timer/counter PVs and SVs			
	Uploading programs	Possible		
	Monitoring I/O memory			
	Monitoring programs			
I/O refreshing		Executed		
Cyclic tasks		Stopped	Tasks set to start when operation starts and tasks that are started with a TASK ON instruction are executed. Other tasks are stopped.	
Interrupt tasks		Stopped	Executed when interrupt conditions are satisfied.	
Outputs from Basic Output Units allocated output bits*1	IOM Hold Bit OFF	OFF immediately after changing to PROGRAM mode.*2	Depends on the program	Depends on the program
	IOM Hold Bit ON	Status held after changing to PROGRAM mode.*2		

*1 When the Output OFF Bit (A500.15) is ON, the outputs from the Basic Output Units will turn OFF regardless of the operating mode and I/O memory status. The outputs will remain OFF even if the power supply is turned ON.

*2 The outputs from Output Units will be refreshed if memory status is changed using Support Software or PT, even in PROGRAM mode.

4

CPU Unit Initialization

This section describes the initialization processing that is performed for the CPU Unit at startup.

4-1 Overview of CPU Unit Initialization 4-2
4-1-1 CPU Unit Initial Settings 4-2
4-2 PLC Setup 4-8
4-3 Creating I/O Tables 4-9
4-3-1 I/O Tables 4-9
4-3-2 Automatic Allocation 4-10
4-3-3 Manual Allocation 4-10
4-4 Setting Routing Tables 4-11
4-4-1 Routing Tables 4-11
4-4-2 Cases in Which Routing Tables Are Required 4-13
4-4-3 Setting and Transferring Routing Tables 4-14
4-5 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units 4-15
4-5-1 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units 4-15
4-5-2 Setting Procedure 4-15
4-6 CPU Bus Unit Setup Area 4-16
4-6-1 CPU Bus Unit Setup Area 4-16
4-6-2 Setting Procedure 4-16

4-1 Overview of CPU Unit Initialization

4-1-1 CPU Unit Initial Settings

Both hardware settings with the DIP switch and rotary switches on the front of the CPU Unit and software settings to set parameters with the CX-Programmer or other software must be made to make the initial settings required for the CPU Unit.

Hardware Settings

● DIP Switch

Set the DIP switch according to the application.

Location	Purpose	Setting method
DIP switch on front of CPU Unit	Main purposes: • Program write protection: Turn ON pin 1. • Automatic transfer at startup: Turn ON pin 2. • Toolbus connection with RS-232C: Turn ON pin 5. • User-defined pin: Turn ON pin 6. • Simple backup: Turn ON pin 7.	Make the settings with the power supply OFF, and then turn the power supply ON.

● Rotary Switches

A unit number and node address are allocated to the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP and CJ2M-CPU3□ in the same way as for a CPU Bus Unit. Words are allocated to the built-in EtherNet/IP port in the CPU Bus Unit Areas in the CIO Area and DM Area according to the unit number setting.

For details on hardware settings, refer to *3-1 CPU Units* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).

Software Settings

The following table gives the software setting applications and setting methods for applicable Units.

Units	Settings		Application	Setting method	File created with Programming Device	Backup destination
CPU Unit	Parameter area	PLC name	Preventing incorrect connection to CPU Units	CX-Programmer	CX-Programmer project file (.CXP)	Built-in flash memory
		PLC Setup	Using non-default specifications			
		Registered I/O tables	Checking for incorrect connection and automatic allocation are required or reserving words for future allocation and other application-specific settings			
		Routing tables	Required when connecting more than one FINS Network Communications Unit (*1)	CX-Integrator	Routing table file (.RTG)	
Ethernet Units		CPU Bus Unit Setup Area (CPU Bus Unit Setups)(*2)	Using Ethernet Units	CX-Programmer	CX-Programmer project file (.CXP)	Built-in flash memory
Controller Link Units (data link tables)			Using user-set data links for Controller Links	CX-Integrator	Data link tables (.CL2)	
FL-net Units			Using FL-net Units	CX-FLnet (FL-net Support Software)	FL-net Support Software setting file (.CSV)	
Special I/O Units and CPU Bus Units	I/O memory	DM Area word allocations (*2) to Special I/O Units or CPU Bus Units	Using Special I/O Units or CPU Bus Units	• CX-Programmer • User program	CX-Programmer project file (.CXP)	Built-in RAM (battery backup)

*1 The following are classified as FINS Network Communications Units.

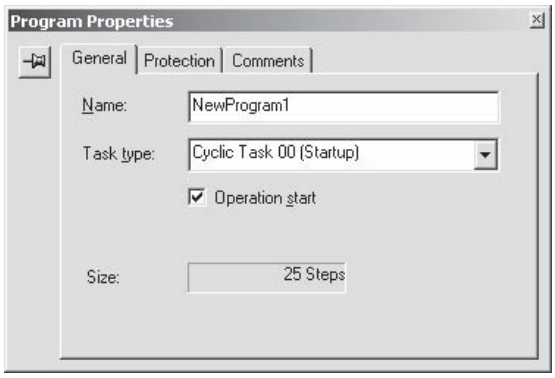
- Network Communications Unit Classified as CPU Bus Unit:
  Controller Link Units, SYSMAC LINK Units, Ethernet Units, DeviceNet Units, and FL-net Units
- Using the CJ2H-CPU6□-EIP or CJ2M-CPU3□ built-in EtherNet/IP port for FINS network communications
- Using a serial gateway with a Serial Communications Unit and using routing tables.

  Routing tables are not applicable to Network Communications Units for Special I/O Units (e.g., CompoNet Master Units).

*2 The initial settings given above for the Special I/O Units or CPU Bus Units are stored in the CPU Unit. Data and programs stored in the Special I/O Units and CPU Bus Units are created separately using Support Software specific to the Unit and then transferred to the Special I/O Units and CPU Bus Units through the CPU Unit.

Additional Information

Setting Program Tasks
The initial settings for assigning programs to tasks is set with the following program properties using the CX-Programmer.



● PLC Name

This is a name that the user sets for the CPU Unit. Make the setting by selecting ***PLC Info*** – ***PLC Info*** from the PLC Menu of the CX-Programmer.

The system will check if the name registered in the PLC matches the PLC name in the project when the CX-Programmer is online.

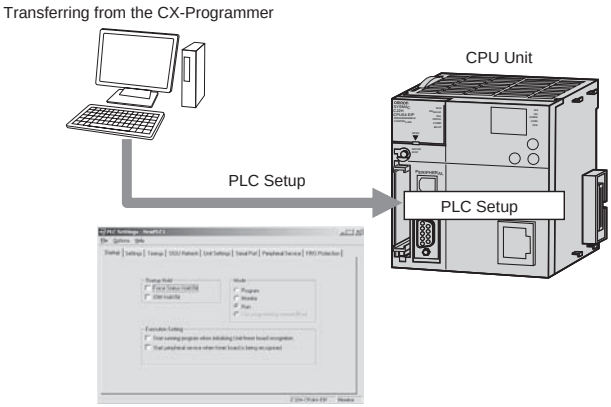
● PLC Setup

The PLC Setup is used to make changes for using the CPU Unit with non-default specifications. The following settings are examples of the defaults for the CPU Unit.

Startup mode: RUN mode
RS-232C port: Host link
Fixed servicing time: 10% of cycle time

To use specifications other than these defaults, change the PLC Setup using the CX-Programmer, and transfer the PLC Setup to the CPU Unit.

⇒ Transferring PLC Setup from the CX-Programmer

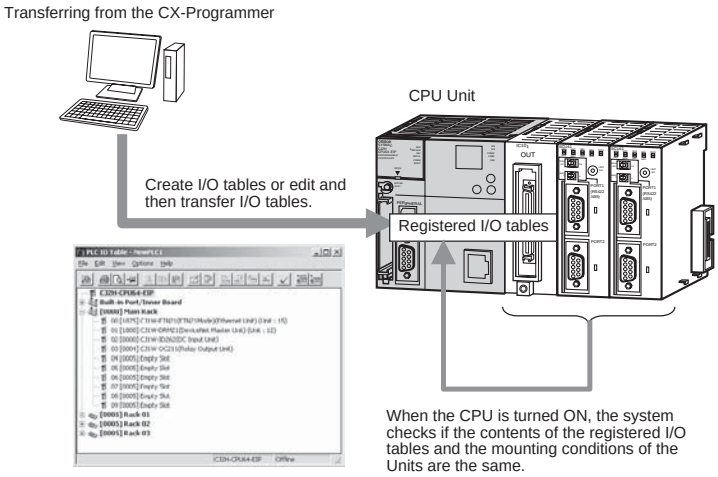


I/O Tables

Create I/O tables to detect incorrect Unit connections when manually setting Unit slots or to manually allocate I/O in the CPU Unit. By default, CPU Unit I/O is automatically allocated in the order that the Units are connected each time the power supply is turned ON.

Create the I/O tables by using one of the following operations from the CX-Programmer.

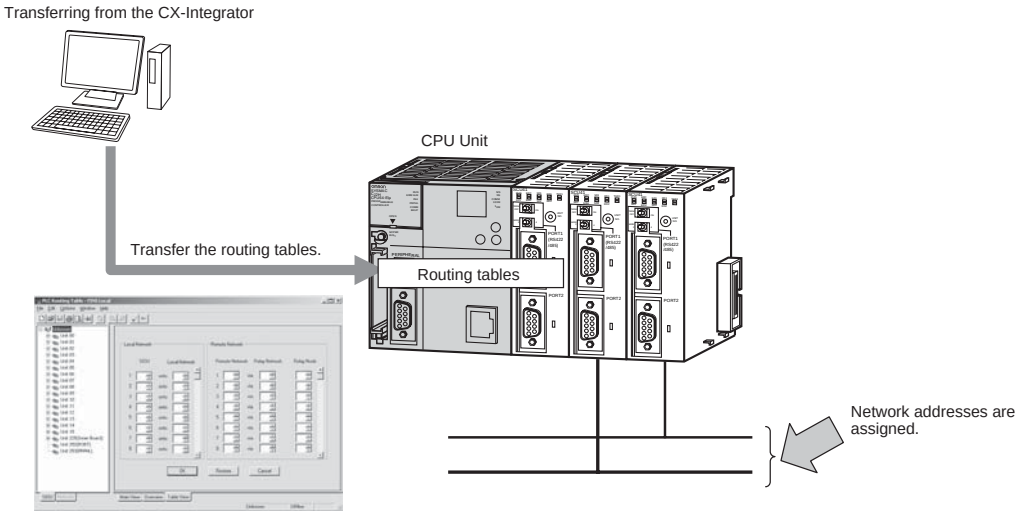
- Online: Perform the procedure for creating the I/O tables with connected Units.
- Offline: Edit the I/O tables (with or without the Units mounted), and then transfer the tables to the CPU Unit.



Routing Tables

Routing tables must be created with the CX-Integrator if more than one FINS Network Communications Unit (CPU Bus Unit) is mounted to the CPU Unit. In the local network table, a network address is assigned to each FINS Network Communications Unit.

Set the local network table with the CX-Integrator and transfer it to the CPU Unit.



CPU Bus Unit Setup Area

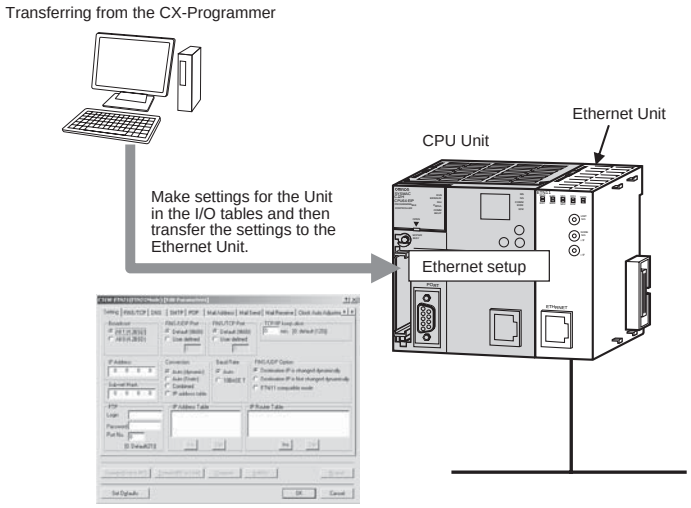
If specific CPU Bus Units, such as Ethernet Units, Controller Link Units, or FL-net Units are used, the particular settings for each of those Units must be made and transferred to the CPU Unit.

Example: Using an Ethernet Unit

Make the Ethernet settings using the CX-Programmer.

To make Ethernet settings, right-click the Ethernet Unit in the I/O tables of the CX-Programmer, and then select *Edit CPU Bus Unit Settings*.

Make the settings for the Ethernet Unit in the I/O tables of the CX-Programmer, and then transfer the I/O tables to the CPU Unit.



Additional Information

If user-set data link tables are to be used with a Controller Link Unit, set the data link tables and then save them in the CPU Unit.

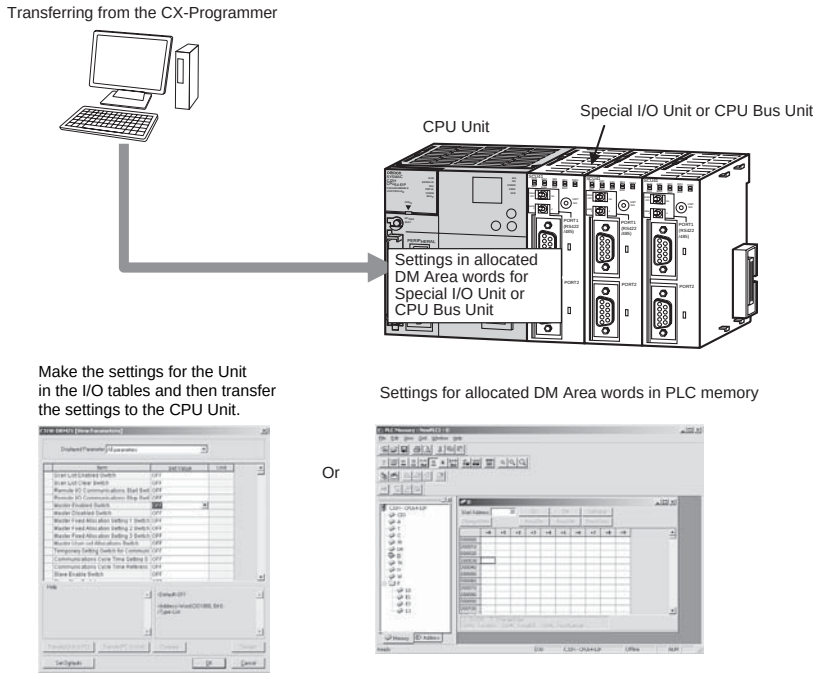
DM Area Word Allocations for Special I/O Units and CPU Bus Units

If a Special I/O Unit or CPU Bus Unit is used, make the settings for the words allocated in the DM Area, and then transfer the settings to the CPU Unit.

Use one of the following methods to set the allocated DM Area words with the CX-Programmer.

- Right-click each Special I/O Unit or CPU Bus Unit in the I/O tables, and then select *Edit CPU Bus Unit Settings* to set the allocated DM Area words.
- Set the allocated DM Area word in PLC memory.

Set the allocated DM Area words for each Special I/O Unit or CPU Bus Unit with the CX-Programmer, and then transfer the settings to the CPU Unit.



4-2 PLC Setup

The PLC Setup contains the basic settings for the CPU Unit. Parameters in the PLC Setup must be changed if the CJ2 CPU Unit is to be used with specifications that are not the defaults. The parameters in the PLC Setup are set by using the CX-Programmer.

For details on the PLC Setup, refer to *Section 9 PLC Setup*.

4-3 Creating I/O Tables

You must create I/O tables only in the following cases. I/O tables are created by using the CX-Programmer.

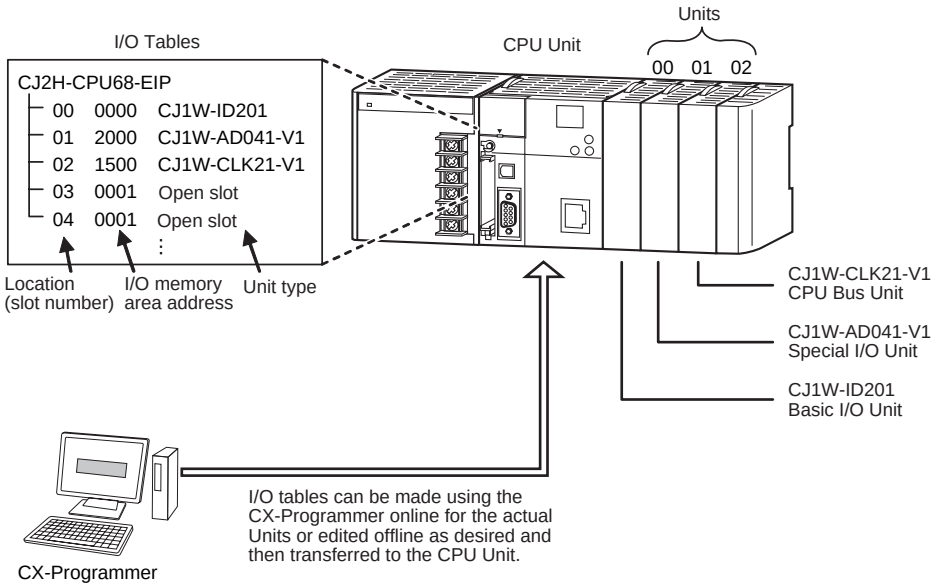
- To check for incorrect connections
- To reserve words

It is not necessary to create I/O tables if checking for incorrect connection is not required and automatic allocation is acceptable.

4-3-1 I/O Tables

The type and location of Units connected to the CPU Unit are registered in the I/O tables. If I/O tables are created, the system will check if the types and locations of the Units actually connected to the CPU agree with the data registered in the I/O tables when the CPU Unit is tuned ON.

The I/O tables can be made based on the actually connected Units when the CX-Programmer is online, or they can be manually set when the CX-Programmer is offline and then transferred to the CPU Unit.



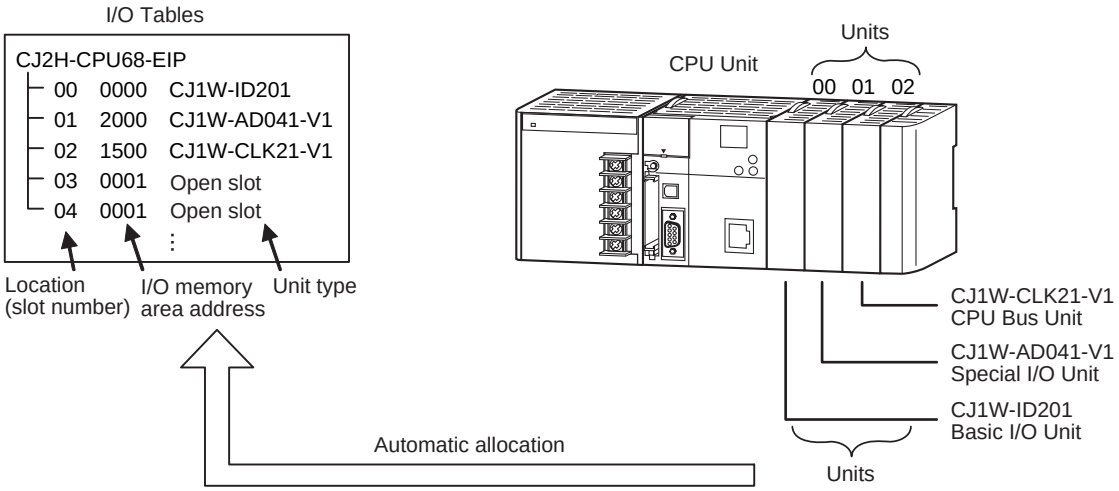
The CPU Unit automatically verifies with the connected Units (actual I/O tables) mounted to it according to this data.

- If there is a mismatch, an I/O setting error will occur, and operation will not start.

4-3-2 Automatic Allocation

With automatic allocation, I/O tables are not created by the user. Every time the power supply is turned ON, I/O memory will be allocated to each Unit based on the locations of the Units. The allocated memory is used for data exchange with the Units.

There are no registered I/O tables, and the system will not check the Unit configuration.



4-3-3 Manual Allocation

With manual allocation, the user creates I/O tables called registered I/O tables.

You can create I/O tables by using one of the following methods with the CX-Programmer.

- Creating I/O Tables Based on the Actual Unit Configuration
  With this method, I/O tables are automatically created based on the configuration of actually mounted Units with the CX-Programmer connected online.
- Editing I/O Tables Manually and Transferring Them to the CPU Unit
  With this method, the I/O tables are manually created offline by registering each Unit in the desired location using the CX-Programmer, and the completed I/O tables are transferred to the CPU Unit.

Whichever method is used, an I/O Setting Error will occur (A401.10 will turn ON) and operation will not start if the configuration of connected Units is different from the unit configuration in the I/O tables when the PLC is turned ON.

For details on I/O allocation methods, refer to *8-1 I/O Allocations*. For details on creating I/O tables, refer to *8-2 Setting CPU Bus Units and Special I/O Units*.

4-4 Setting Routing Tables

Settings for routing tables must be made with the CX-Integrator if more than one FINS Network Communications Unit is mounted to the PLC and the following operations are to be performed.

- The network that is being accessed is switched from Support Software or an instruction in a ladder program.
- Communications are performed across network layers.

4-4-1 Routing Tables

A communications method called FINS is used between OMRON Network Communications Units (CPU Bus Units). The transmission path, however, is not specified in the FINS protocol. Routing tables define the network address settings and communications paths required when FINS network communications are performed.

Applicable Units

Network addresses for the following Units are set in the routing tables.

- Network Communications Units for CPU Bus Units:
  Controller Link Units, SYSMAC LINK Units, Ethernet Units, EtherNet/IP Units used for FINS network communications, DeviceNet Units, and FL-net Units
- Built-in EtherNet/IP port on the CJ2H-CPU6□-EIP and CJ2M-CPU3□ when used for FINS network communications
- Serial ports on Serial Communications Units when communications are performed across network layers using serial gateway

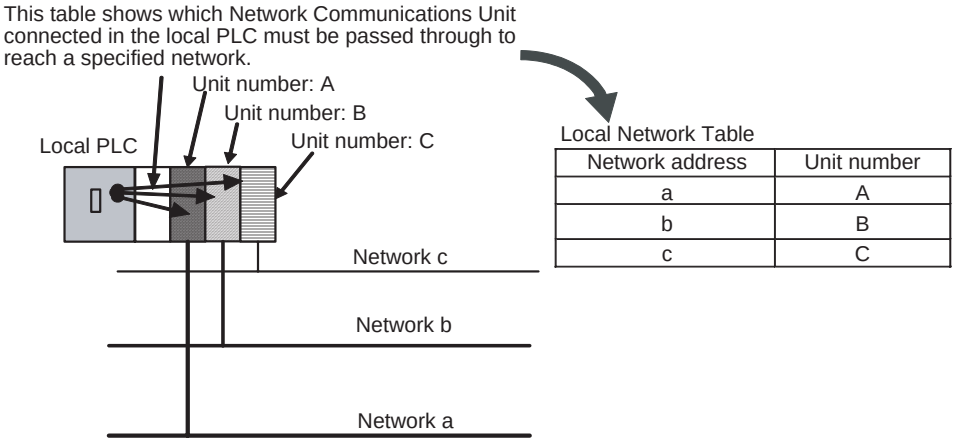
Network Communications Units classified as Special I/O Units (e.g., CompoNet Master Units) do not need to be set in the routing tables.

Tables Comprising Routing Tables

Routing tables consist of the following two tables.

● Local Network Table

Network addresses from 1 to 127 are set for the Network Communications Units (CPU Bus Units) connected in the local PLC. The local network table is used by the CPU Unit to identify Network Communications Units mounted to the local PLC and the corresponding communications networks if more than one Network Communications Unit is mounted to the PLC.

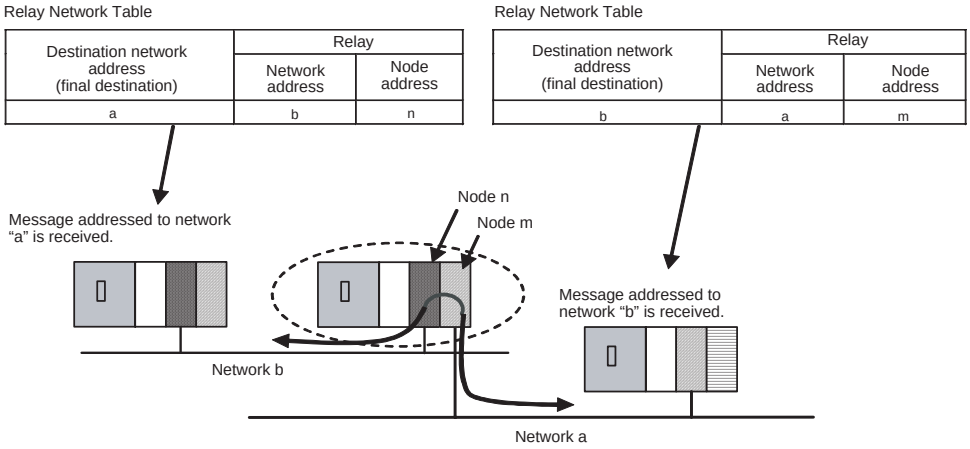


The local network table gives the unit number of the Network Communications Unit connected in the local PLC and the corresponding address of the FINS network connected to that Unit for FINS communications.

● Relay Network Table

A relay network table is set to perform FINS communications across network layers. The relay network table tells how to relay communications until the message reaches the final destination. The table gives the network address and node address of the first relay point (i.e. first point to reach) on the route to a destination network (final network) to which the local PLC is not connected.

The destination network is reached by progressing through the relay points.



4-4-2 Cases in Which Routing Tables Are Required

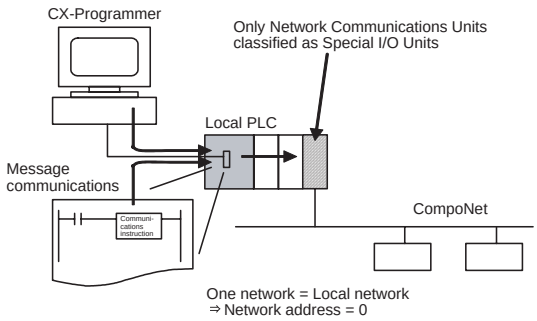
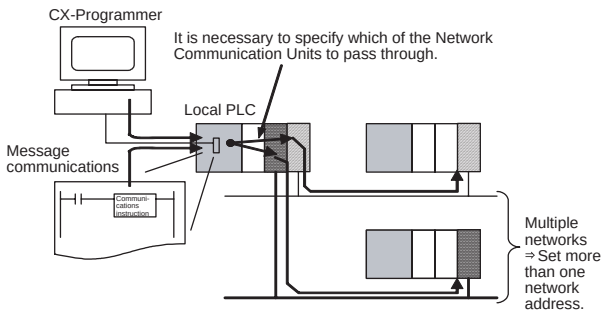
Set the routing tables (local network table and relay network table) in the CPU Unit as required by the application.

Precautions for Correct Use

Conditions Required for a Local Network Table

A local network table is required if there is more than one Network Communications Unit connected in the PLC even when relaying is not performed.

The following table shows whether routing tables are required or not according to the network usage conditions.

Network usage condition	System configuration	Routing tables	
		Local network table	Relay network table
A Network Communications Unit is not connected in the PLC or only a Network Communications Unit classified as a Special I/O Unit (e.g., CompoNet) is connected.*1		Not required	Not required
More than one Communications Unit classified as CPU Bus Units is connected in the PLC.		Required for all PLCs in the networks*2	Required if communications will be performed across network layers. *3

*1 Set the remote network address to 0 if a node in the network will be accessed using a Programming Device.

*2 If there is no local network table and access is made with the network address set to 0 using a Programming Device or an instruction in a ladder program, the network with the Network Communications Unit with the lowest unit number will be accessed automatically in FINS communications. In this way, the network can be accessed by setting the lowest unit number for the Network Communications Unit in the network to be accessed without setting local network tables.

*3 Relay network tables do not need to be set if communications will not be performed across network layers.

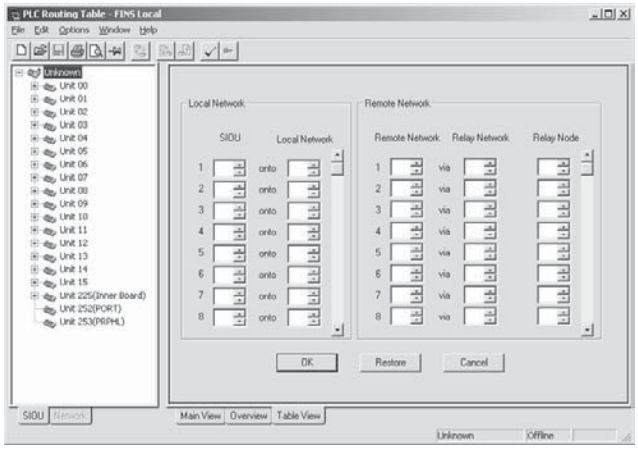
4-4-3 Setting and Transferring Routing Tables

1 Start the CX-Integrator.

2 Select *Start Routing Table* from the Tools Menu.

3 Select *FINS Local*.

4 Edit the routing tables on the Table View Tab Page.



5 Connect the CX-Integrator online, and then select *Transfer to PLC from* the Options Menu to transfer the routing tables to the CPU Unit.

6 Select *Save Local Routing Table File* from the File Menu.

Precautions for Correct Use

Routing Table Data File
The routing tables are stored in an individual file (.rtg) created with the CX-Integrator. It is not included in the CX-Programmer project file (.cxp).

4-5 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units

These settings must be made if Special I/O Units or CPU Bus Units are used. Set the DM Area words allocated to Special I/O Units and CPU Bus Units using the CX-Programmer.

4-5-1 Setting Allocated DM Area Words for Special I/O Units and CPU Bus Units

The initial settings for Special I/O Units and CPU Bus Units are made words allocated to the Unit in the DM Area of the I/O memory in the CPU Unit. The settings in the allocated DM Area words are transferred to the Special I/O Units and CPU Bus Units from the CPU Unit when the power supply is turned ON.

The allocated DM Area words are allocated from the following addresses

● Initial Settings for Special I/O Units

One hundred words per Unit are allocated from between D20000 and D29599 according to the unit number set on the rotary switches on the front of the Unit.

● Initial Settings for CPU Bus Units

One hundred words per Unit are allocated from between D30000 to D31599 according to the unit number set on the rotary switches on the front of the Unit.

The actual settings and transfer timing for the allocated DM words depend on the type of Special I/O Unit and CPU Bus Unit and area. For details, refer to the operation manual for the Special I/O Unit or CPU Bus Unit.



Additional Information

Settings for Special I/O Units and CPU Bus Units are also contained in the CPU Bus Unit Setup Area for specific Units, such as Ethernet Units and Controller Link Units, and data and programs are also stored in Special I/O Units and CPU Bus Units (created using the Support Software for the specific Unit).

4-5-2 Setting Procedure

Make the settings using one of the following methods.

- Select the Units in the PLC I/O Table Window of the CX-Programmer, make the settings for the Special I/O Units and CPU Bus Units, and then transfer the settings to the CPU Unit.
- Set the data for individual addresses in the allocated DM Area words in the PLC Memory Window of the CX-Programmer, and then transfer the settings to the CPU Unit.
- Set the data for individual addresses in the allocated DM Area words by using MOV(021) or other instructions in the user program.

Refer to *8-2 Setting CPU Bus Units and Special I/O Units* for the procedure for allocating DM Area words to CPU Bus Units and Special I/O Units with the CX-Programmer.

4-6 CPU Bus Unit Setup Area

4-6-1 CPU Bus Unit Setup Area

A setup area stores the settings for specific CPU Bus Units mounted to the CPU Unit. The following three types of Units and settings use this area.

- Ethernet Units: Ethernet settings
- Controller Link Units: Data link tables (when user-set tables are used)
- FL-net Units: FL-net settings

4-6-2 Setting Procedure

Make the settings using the following Support Software.

Data	Support Software
Ethernet Unit settings	Make the settings for the Special I/O Unit or CPU Bus Unit in the I/O tables of the CX-Programmer or make the setting using HTML.
Controller Link Unit data link tables	Make the settings using data link tables in the CX-Integrator.
FL-net Unit settings	Make the settings using the CX-FLnet FL-net Support Software.

Memory Used for CPU Bus Unit Setup Area

The maximum amount of memory that can be used for the CPU Bus Unit Setup Area is 10,752 bytes. Design the system so that the memory used for the CPU Bus Unit Setup Area is within the limit according to the combination of CPU Bus Units in the PLC. If the limit is exceeded, some Units may operate only at the default settings, and some may not operate.

● Memory Used for CPU Bus Unit Setup Area

Unit	Model	Size used (bytes)
Controller Link Unit	CJ1W-CLK21-V1	512
Ethernet Unit	CJ1W-ETN11	412
	CJ1W-ETN21	994
FL-net Unit	CJ1W-FLN22	988
EtherNet/IP Unit	CJ1W-EIP21	0
Serial Communications Unit	CJ1W-SCU21/31/41-V1 CJ1W-SCU22/32/42	
DeviceNet Unit	CJ1W-DRM21	
Position Control Unit	CJ1W-NCF71	
Motion Control Unit	CJ1W-MCH71	
Storage and Processing Unit	CJ1W-SPU01-V2	

Units that use 0 bytes do not use the CPU Bus Unit Setup Area.

5

Understanding Programming

This section describes the basics of programming CJ2 CPU Units.

5-1 Programming 5-3
5-1-1 Programming Overview 5-3
5-1-2 Basic Ladder Diagram Concepts 5-6
5-1-3 ST Language 5-8
5-1-4 SFC Overview 5-9
5-2 Tasks 5-11
5-2-1 Overview of Tasks 5-11
5-2-2 Cyclic Tasks 5-14
5-2-3 Interrupt Tasks 5-20
5-2-4 Designing Tasks 5-28
5-3 Sections 5-38
5-3-1 Overview of Sections 5-38
5-4 Function Blocks 5-40
5-4-1 Function Blocks 5-40
5-4-2 Features of Function Blocks 5-41
5-4-3 Function Block Specifications 5-42
5-5 Symbols 5-45
5-5-1 Overview 5-45
5-5-2 Types of Symbols 5-46
5-5-3 Global Symbols 5-48
5-5-4 Local Symbols 5-48
5-5-5 Network Symbols (CJ2H-CPU6@-EIP and CJ2M-CPU3@ Only) 5-49
5-5-6 Variables in Function Blocks 5-53
5-5-7 Symbol Data Types 5-54
5-5-8 Automatic Address Allocation to Symbols 5-59
5-6 Instructions 5-60
5-6-1 Basic Understanding of Instructions 5-60
5-6-2 Specifying Operands 5-67
5-6-3 Data Formats 5-75
5-6-4 I/O Refresh Timing 5-79

5-7 Index Registers 5-84
5-7-1 What Are Index Registers? 5-84
5-7-2 Using Index Registers 5-84
5-7-3 Processing Related to Index Registers 5-88
5-7-4 Monitoring Index Registers 5-89
5-7-5 Sharing Index and Data Registers between Tasks 5-90
5-8 Specifying Address Offsets 5-92
5-8-1 Overview 5-92
5-8-2 Examples of Address Offset Application 5-95
5-9 Checking Programs 5-96
5-9-1 Errors during CX-Programmer Input 5-96
5-9-2 Program Checks with the CX-Programmer 5-96
5-9-3 Debugging with the Simulator 5-97
5-9-4 Program Execution Check 5-100
5-10 Precautions 5-103
5-10-1 Condition Flags 5-103
5-10-2 Special Program Sections 5-108

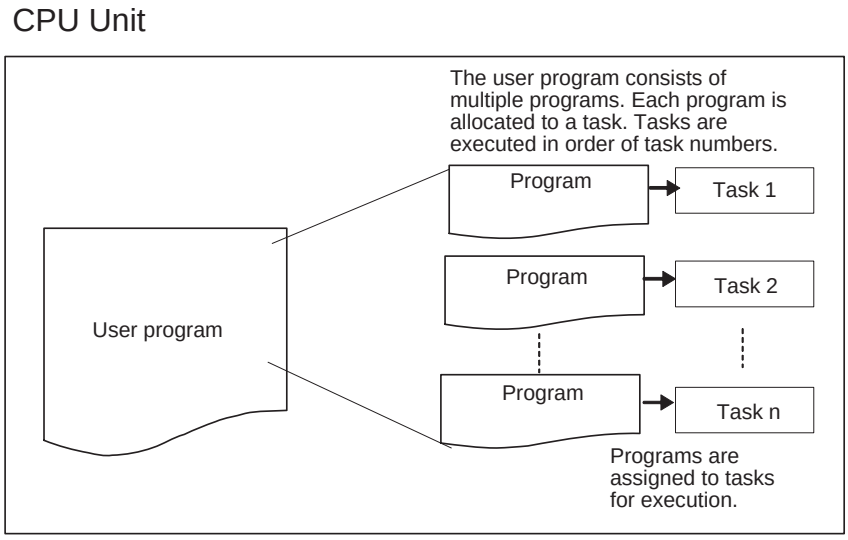
5-1 Programming

5-1-1 Programming Overview

Structure of User Programs

User programs are created by using the CX-programmer. Programs consist of the following parts.

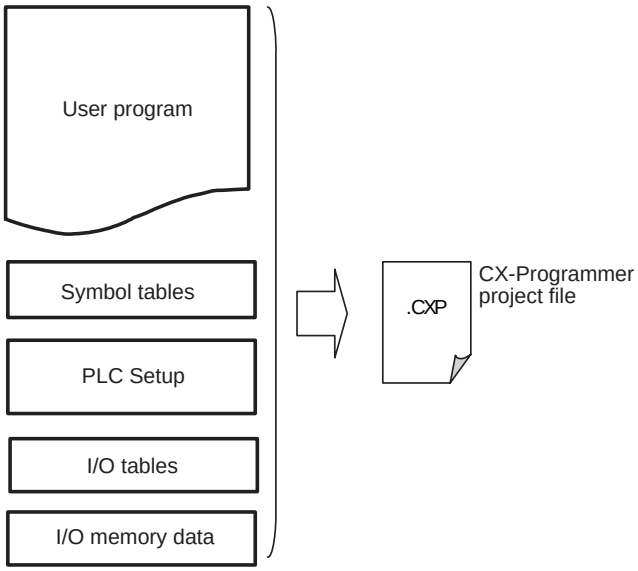
- Programs
  The entire user program consists of multiple programs. Each program ends with an END(001) instruction.
- Tasks (Smallest Executable Unit)
  A program is assigned to a task to execute it. (In the CX-Programmer, the task number is specified in the properties as a program attribute.) Tasks include cyclic tasks (executed with normal cyclic processing), interrupt tasks (executed when interrupt conditions have been completed), scheduled interrupt tasks (executed at specified intervals), and the power OFF interrupt task (executed when the power is interrupted).



- Sections
  When creating and displaying programs with the CX-Programmer, the one program can be divided into any number of parts. Each part is called a section. Sections are generally made to make the program easier to understand.
- Subroutines and Function Blocks
  In one program, you can create subroutine programs and function blocks.

User Program Data

The entire user program is saved in a CX-Programmer project file (.CXP) with other parameters, such as symbol tables, PLC Setup data, I/O tables, and I/O memory data.

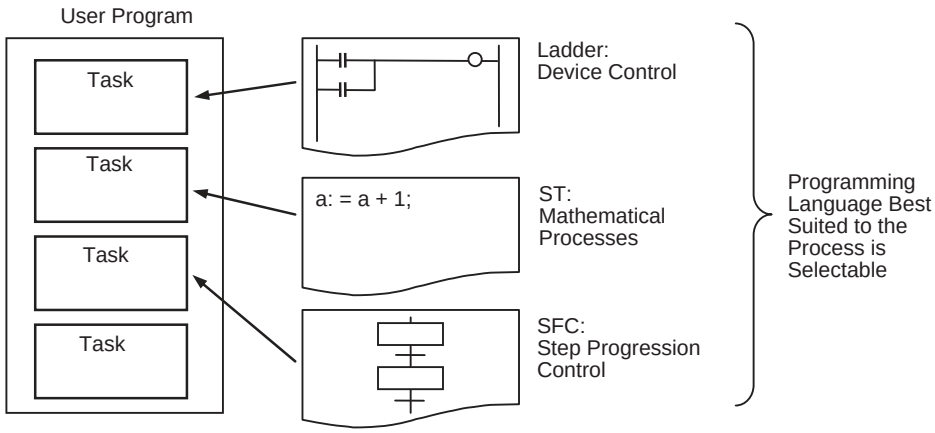


Programming Languages

Programs can be written using any of the following three programming languages.

- Ladder diagrams
- Structured text (ST)
- Sequential function charts (SFC)

Programs written in ladder diagrams, ST, or SFC assigned to tasks to execute them.



You can use the best language for each process by using different languages for different purposes, such as ladder programming for the device control or ST for mathematical processing.

Each language can be used in the following locations.

Location / Language	Tasks	Action programs and transition programs in SFCs	Algorithms in function block definitions
Ladder diagrams	Yes	Yes	Yes
ST	Yes	Yes	Yes
SFC	Yes	---	---

The following addressing methods can be used for each language.

Addressing method / Language	Physical addresses	Symbols
Ladder programming	Yes	Yes
ST	Depends*1	Yes
SFC	Depends*2	Depends*2

*1 Physical addresses can be used by assigning them to symbols.

*2 Physical address and symbols can be used in Boolean actions and transitions in SFC charts.

Program Capacity

The maximum program capacities of the CJ2 CPU Units for all user programs (i.e., the total capacity for all tasks) are given in the following table.

Model	Program capacity	I/O capacity
CJ2H-CPU68(-EIP)	400K steps	2,560 points
CJ2H-CPU67(-EIP)	250K steps	
CJ2H-CPU66(-EIP)	150K steps	
CJ2H-CPU65(-EIP)	100K steps	
CJ2H-CPU64(-EIP)	50K steps	
CJ2M-CPU□5	60K steps	
CJ2M-CPU□4	30K steps	
CJ2M-CPU□3	20K steps	
CJ2M-CPU□2	10K steps	
CJ2M-CPU□1	5K steps	

It is possible to check the program capacity by selecting *View* – *Memory View* in the CX-programmer.

The size of a ladder instruction depends on the specific instruction and operands that are used. For details, refer to *A-2 Instruction Execution Times and Number of Steps*.

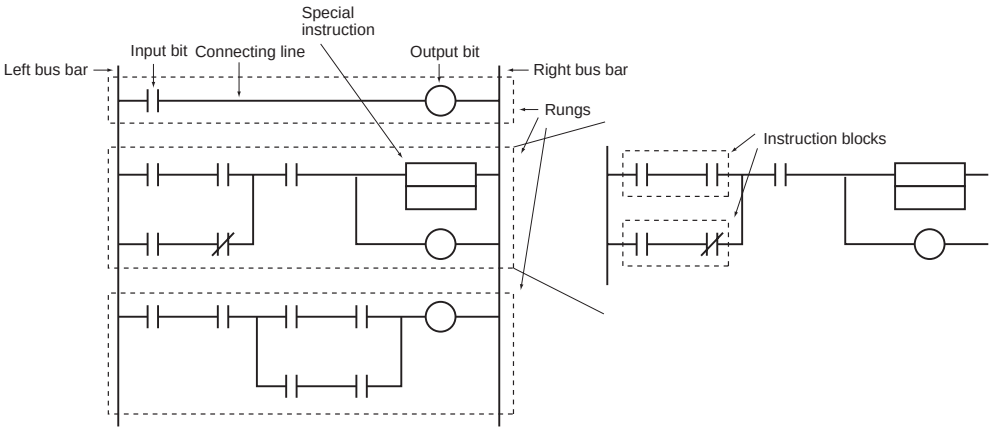
Refer to *5-4-3 Function Block Specifications* for the amount of memory used for function blocks.

5-1-2 Basic Ladder Diagram Concepts

Ladder diagram logic is a basic language for PLCs that is written in a form that appears similar to electrical circuits. Instructions are executed in the order they are recorded in memory (mnemonic order). It is important that you correctly understand the basic programming concepts as well as the execution order.

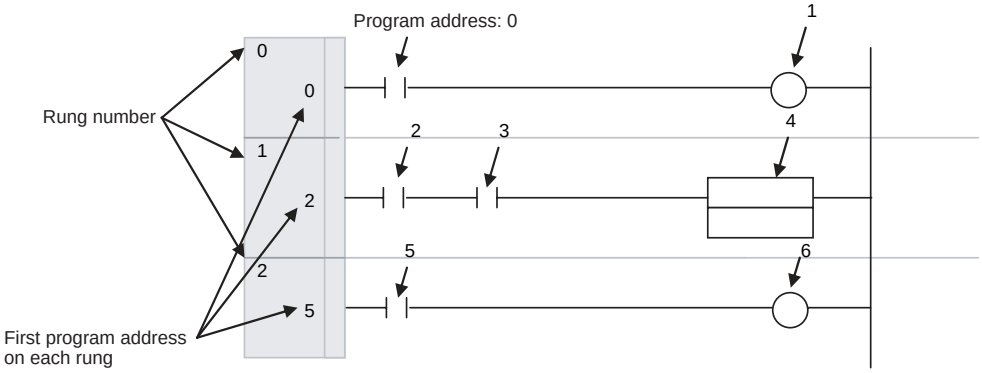
● General Structure of the Ladder Diagram

A ladder diagram consists of left and right bus bars, connecting lines, input bits, output bits, and special instructions. A program consists of one or more program runs. A program rung is a unit that can be partitioned when the bus is split horizontally. In mnemonic form, a rung is all instructions from a LD/LD NOT instruction to the output instruction just before the next LD/LD NOT instructions. A program rung consists of instruction blocks that begin with an LD/LD NOT instruction indicating a logical start.



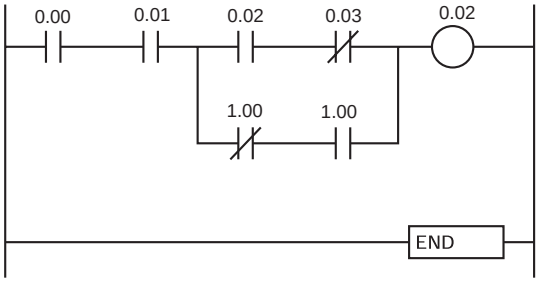
Additional Information

With the CX-Programmer, the rung number and first program address on each rung are displayed at the left of each rung.



Mnemonics

It has program addresses, and one program address is equivalent to one instruction. Program addresses contain six digits starting from 0.



Program Address	Instruction (Mnemonic)	Operand
0	LD	0.00
1	AND	0.01
2	LD	0.02
3	ANDNOT	0.03
4	LDNOT	1.00
5	AND	1.01
6	OR LD	
7	AND LD	
8	OUT	2.00
9	END	

5-1-3 ST Language

The ST (Structured Text) language is a high-level language code for industrial controls (mainly PLCs) defined by the IEC 61131-3 standard. The standard control statements, operators, and functions make the ST language ideal for mathematical processing that is difficult to write in ladder programming. (The

ST language does not support all of the processing that can be written in ladder diagrams. The ST language that conforms to the IEC 61131-3 standard is supported.



Additional Information

For details on ST programming specifications, notation, and input procedures, refer to the *CX-Programmer Operation Manual: Function Blocks and Structured Text* (Cat. No. W447).

● Features of ST Programming

- There are many control statements available, such as loop statements and IF-THEN-ELSE statements, many operators such as arithmetic operators, comparison operators, and AND/OR operators, as well as many mathematical functions, string extract and merge functions, Memory Card processing functions, string transfer functions, and trigonometric functions.
- Programs can be written like high-level languages such as C, and comments can be included to make the program easy to read.
- ST programs can be uploaded and downloaded just like ordinary programs, but ST program tasks cannot be uploaded and downloaded in task units.
- Function blocks (ladder or ST language) can be called in ST programs.
- One-dimensional array variables are supported for easier data handling in applications.

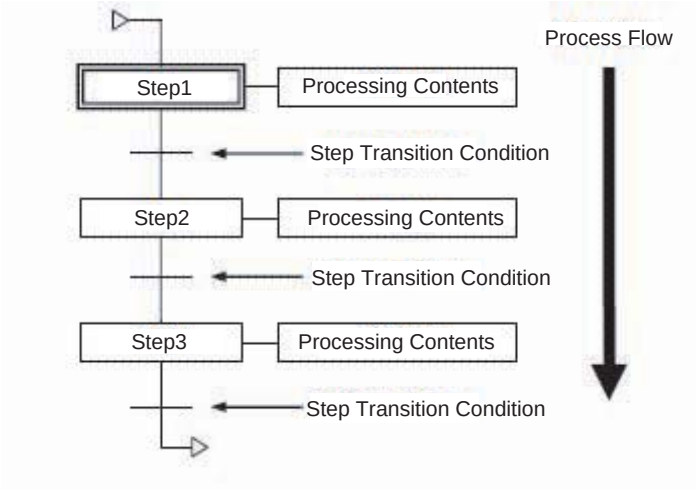
● Specifications

Item	Specification
ST program units	• Tasks • Algorithms for function block definitions • Action programs and transition programs in SFCs.
Address specifying procedure	Addresses are specified using symbols. Any address can be assigned to the symbol.
Force-setting and force-resetting	Supported.
Online editing	ST program editing

5-1-4 SFC Overview

The SFC (Sequential Function Chart) language is a graphical programming language developed to facilitate the description of step progression programs, which mainly control sequential processes.

SFC, with its graphical representation of step flow and with description of the conditions for step progression and the actions in each step, allows users to program the control of sequential processes.



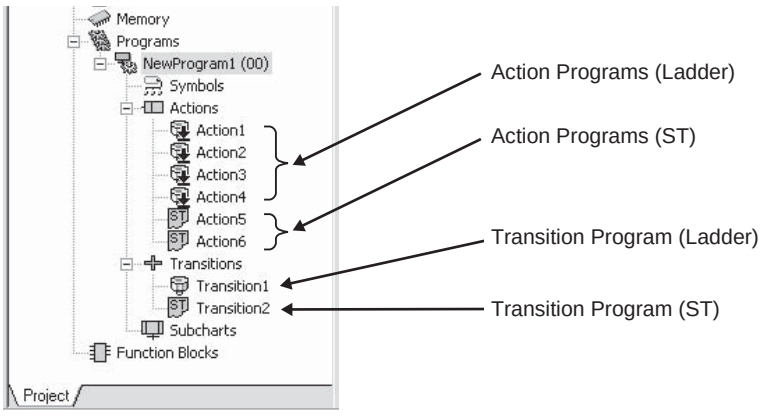
Additional Information

For details on SFC programming specifications, notation, and input procedures, refer to the *CX-programmer Operation Manual: SFC Programming* (Cat. No. W469).

● Features of SFC Programming

Ladder Diagrams and ST as Part of SFC Programs

The step actions and transition conditions within an SFC program can be coded in either ladder diagrams or ST.



● Simultaneous Display/Editing of SFC Charts and Action/Transition Programs

Online Editing and Debugging

SFC charts can be edited online. Furthermore, action blocks can be hidden while debugging, and the step progression status can be checked. The SFC chart editor will display the action programs in the program view, even while the action blocks are hidden.

● Specifications

Item	Specification
SFC program unit	Tasks (1 task = 1 SFC chart)
Tasks supported for SFC program allocation	Cyclic or extra cyclic tasks
SFC elements	Steps, transitions, actions, jumps, subcharts Note: SFC elements are automatically registered to local variables.
Address specifying method	A symbol or a physical address can be specified for a Boolean action or transition in an SFC. An physical address cannot be specified, however, if ST programming is used.
Force-setting and force-resetting	Transitions can be force-set/force-reset online, for a step-by-step execution.
Online editing	Enables SFC editing, action addition/deletion, and transition addition/deletion.

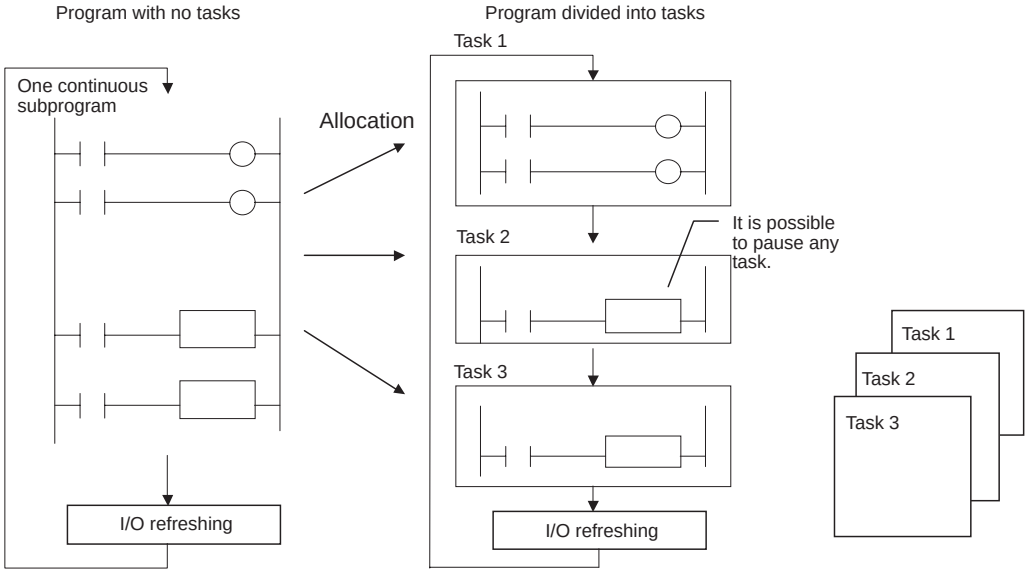
5-2 Tasks

5-2-1 Overview of Tasks

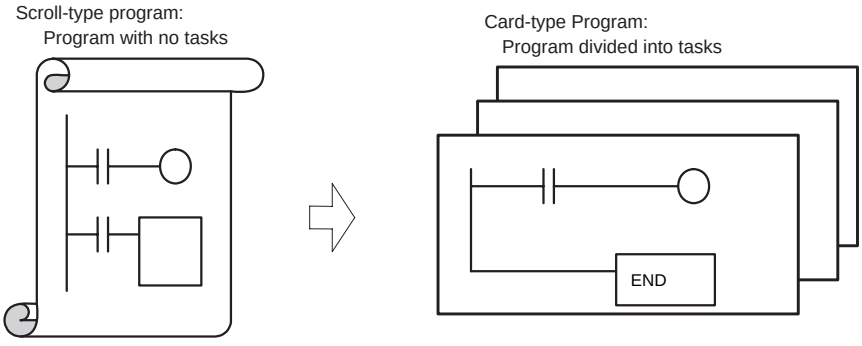
Concept of Tasks

Tasks are used to divide a program into large units and specify the order in which to execute each unit or programming. Instructions in any one task can be used to enable or disable the execution of other tasks.

This means that several program components can be assembled as different tasks, and that only specific tasks can then be executed as needed for the current product model or process being performed. This enables switching between different tasks for different processes. Therefore performance (cycle time) is greatly improved because only required programs will be executed as needed.



A program without tasks is like reading an entire scroll form the beginning, while a program with tasks is like reading cards individually. Each card can enable or disable other cards. Reading disabled cards will be skipped.



The entire program is read.

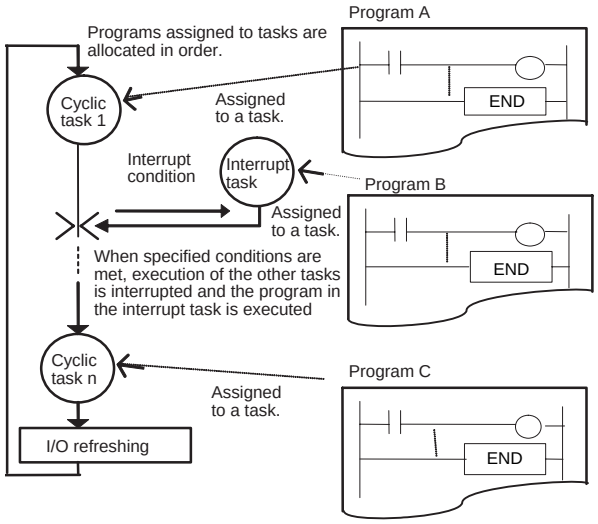
Only the required cards are read.

There are basically two types of tasks.

Type of task	Description	Applicable programming language	Execution conditions
Cyclic tasks	Executed once per cycle	Ladder diagrams SFC ST	Executed when one of the following methods is used to place the task into READY status. • Setting the *Activated at the start of operation* property using the CX-Programmer (program property setting) • Turning ON the task by using a Task Control Instruction
Interrupt tasks*1,*2	Executed when a specific condition occurs. The process being executed is interrupted.	Ladder diagrams ST	An interrupt task is placed into READY status when the interrupt conditions that is set for it occurs. Conditions can be set for each of the following interrupt tasks. • Power OFF interrupt tasks • Scheduled interrupt tasks • I/O interrupt tasks • External interrupt tasks • Built-in input interrupt tasks (CJ2M CPU Units only)

*1 Cyclic execution (i.e., execution once per cycle) can be performed for an interrupt task just as with cyclic tasks by using Task Control Instructions to turn ON the interrupt task. (These tasks are called extra cyclic tasks.)

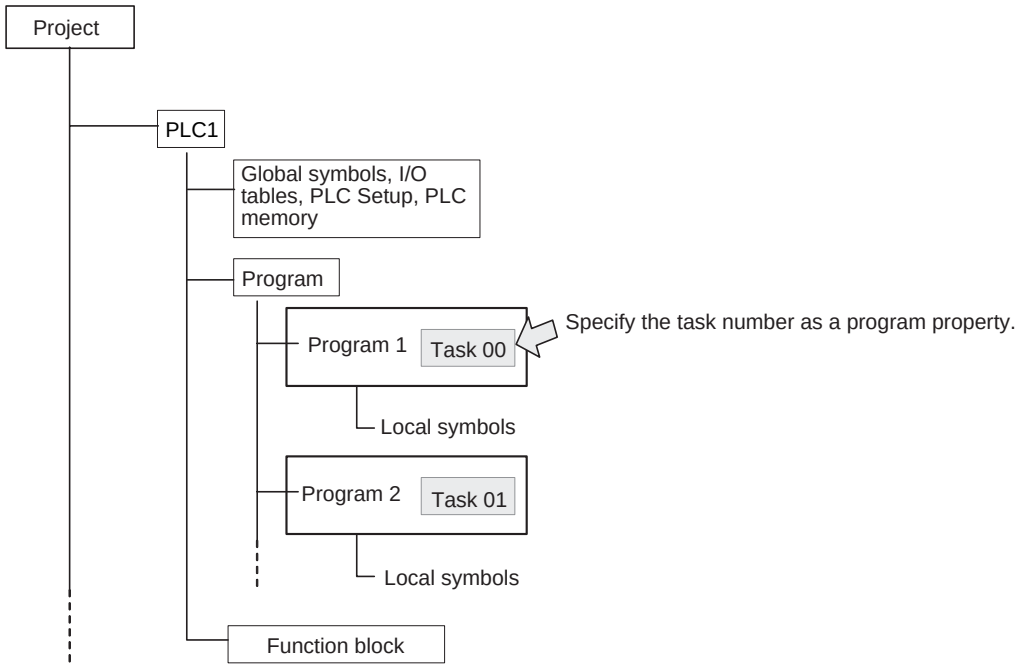
*2 Do not use SFC programs in interrupt tasks. They will not execute normally. SFC programs can be used, however, if the interrupt tasks are executed as extra cyclic tasks.



CX-Programmer Operations for Tasks

Set the task numbers for programs as program properties using the CX-Programmer.

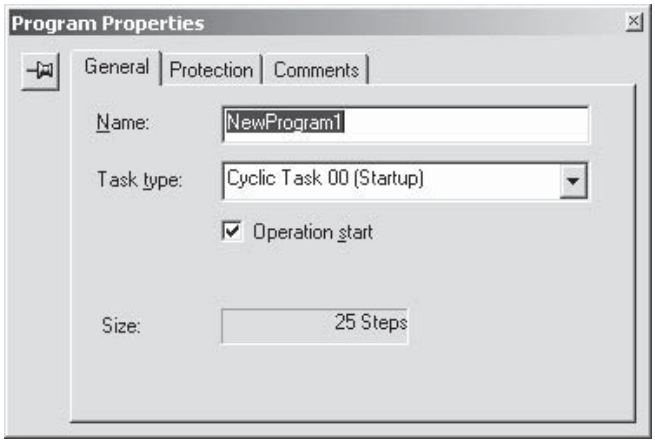
● Settings for Task Numbers



Specify the task type and number as attributes for each program.

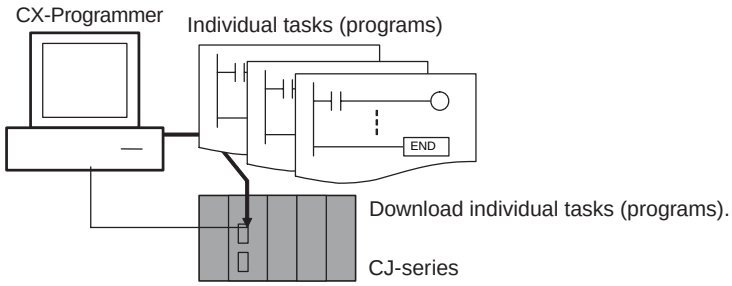
1 Select ***View/Properties***, or click the right button and select **Properties** on the popup menu, to display the program that will be allocated a task.

2 Select the **General** tab, and select the **Task Type** and **Task No.** For a cyclic task, select the *Operation start* Check Box if you want to start executing the task when operation is started.



● Downloading and Uploading Individual Tasks

Individual program tasks can be uploaded and downloaded from the CX-Programmer.



● Restrictions to Function Block Use

Individual tasks cannot be downloaded for programs containing function blocks (uploading is possible).

● Restrictions to Using SFC Programs and ST Programs

Tasks to which SFC programs or ST programs are assigned cannot be uploaded or downloaded individually.

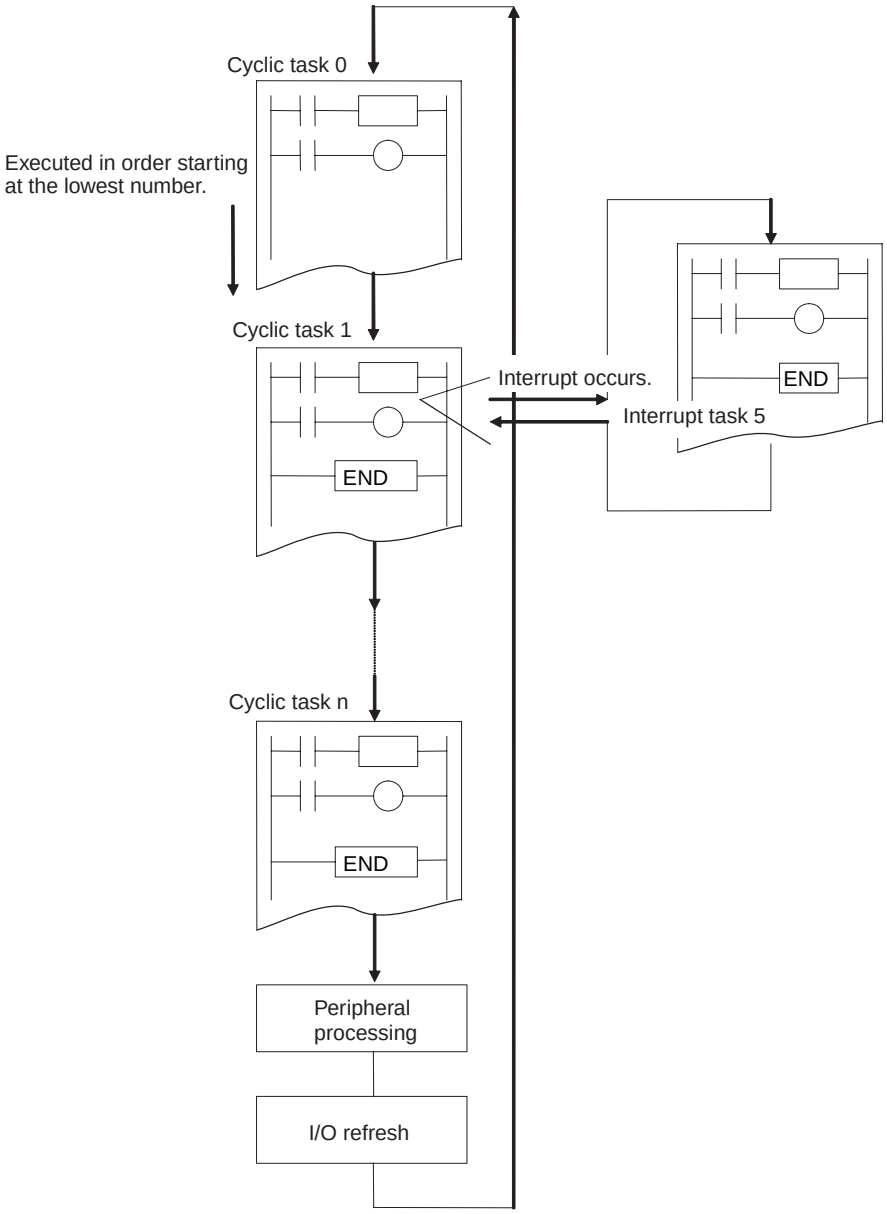
5-2-2 Cyclic Tasks

Cyclic Tasks

Cyclic tasks are executed once per cycle in order starting with the lowest task number. Up to 128 tasks can be used (cyclic task numbers 0 to 127). The tasks can be started by setting the *Activated at the start of operation* Property using the CX-Programmer or by using Task Control Instructions. For information on Task Control Instructions, refer to *A-2 Instruction Execution Times and Number of Steps*. If extra cyclic tasks are used, a maximum of 384 cyclic tasks can be used.

Precautions for Correct Use

Do not use the Task Control Instructions (TKON(820) and TKOFF(821)) to control tasks containing SFC programs. Otherwise they will not execute normally.



Additional Information

- All Condition Flags (ER, CY, Equals, AER, etc.) and instruction conditions will be cleared at the beginning of a task. Therefore, Condition Flags cannot be read between two tasks.
- Interlocks (e.g., IL and ILC instructions), jumps (e.g., JMP, CJP, and JME instructions), and subroutines (e.g., SBS, RET, and SBN instructions) must be completed within each individual task. For example, jumping cannot be performed from one task to another. If subroutines will be used by more than one tasks, use global subroutines (GSBS(750), GRET(752), or GSBN(751) instructions).

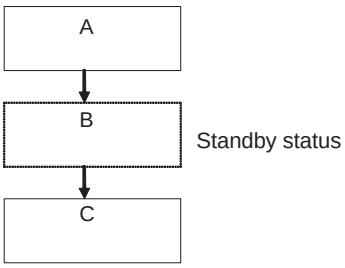
Cyclic Task Status

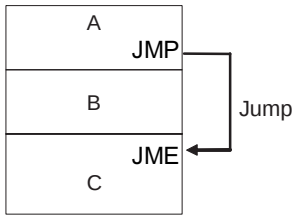
● READY Status

Cyclic tasks are executed in order starting with the lowest task number. Tasks for which the *Activated at the start of operation* Property is set will be executable when the operating mode is changed from PROGRAM mode to RUN or MONITOR mode. A TASK ON instruction can be used to change the status from STANDBY status to READY status. If a TASK OFF instruction is used to stop the local task, the task will not be executed beyond the TASK OFF instruction.

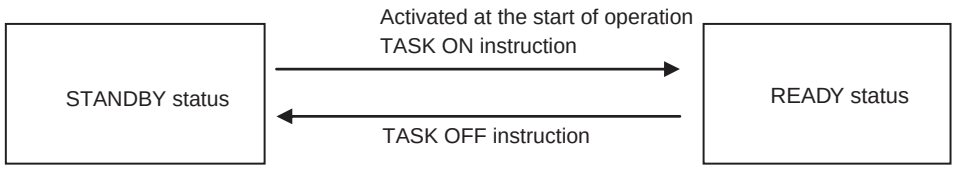
● STANDBY Status

Instructions will not be executed for tasks in STANDBY status. Using a TASK OFF instruction will change the status from READY status to STANDBY status. The cycle time can be shortened by dividing the overall user program into tasks then starting and stopping the tasks with the Task Control Instructions. The program can be divided into tasks according to when different parts of the overall program need to be executed.





● Status Transitions



Additional Information

STANDBY status functions exactly the same way as a jump (JMP-JME). Output status from a STANDBY task will be maintained.

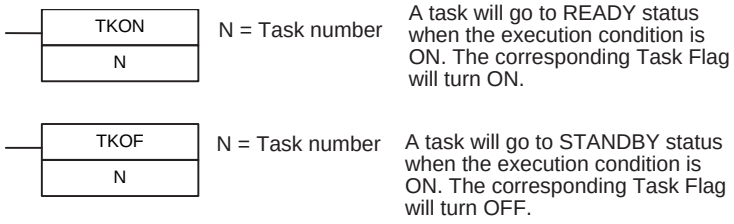
Using Cyclic Tasks

● Task Control Instructions

Use the Task Control Instructions to start or stop a cyclic task from a program. The instructions used to start and stop tasks depend on the programming language as shown in the following list.

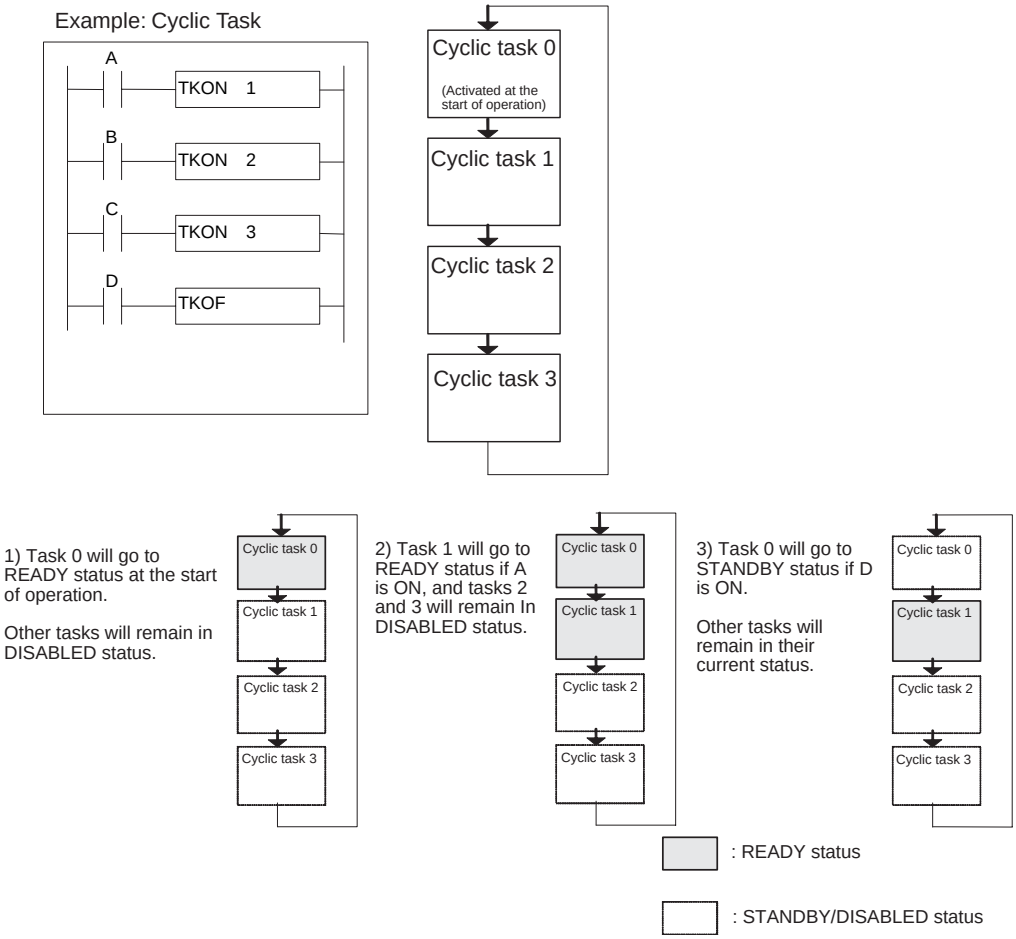
- Ladder programming and ST programming: TKON(820) (TASK ON instruction) and TKOF(821) (TASK OFF instruction)
- SFC programming: SFCON(789) (SFC ON instruction) and SFCOFF(790) (SFC OFF instruction)

- TASK ON and TASK OFF Instructions



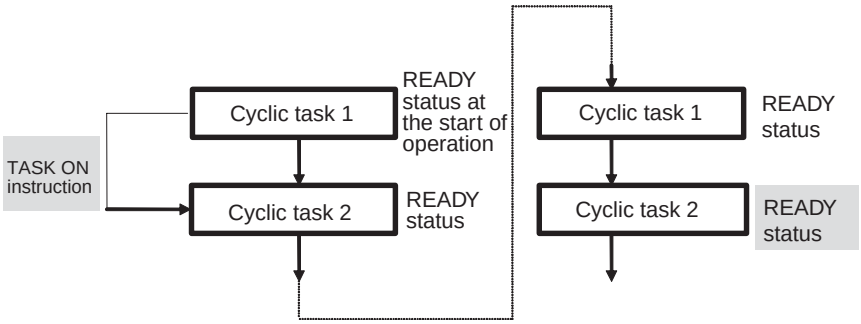
The TASK ON and TASK OFF instructions can be used to change any cyclic task between READY and STANDBY status at any time. A cyclic task that is in READY status will maintain that status in subsequent cycles, and a cyclic task that is in STANDBY status will maintain that status in subsequent cycles. The TASK ON and TASK OFF instructions can be used only in cyclic tasks. They cannot be used in interrupt tasks.

Note At least one cyclic task must be in READY status in each cycle. If there is no cyclic task in READY status, the Task Error Flag (A295.12) will turn ON, and the CPU Unit will stop.

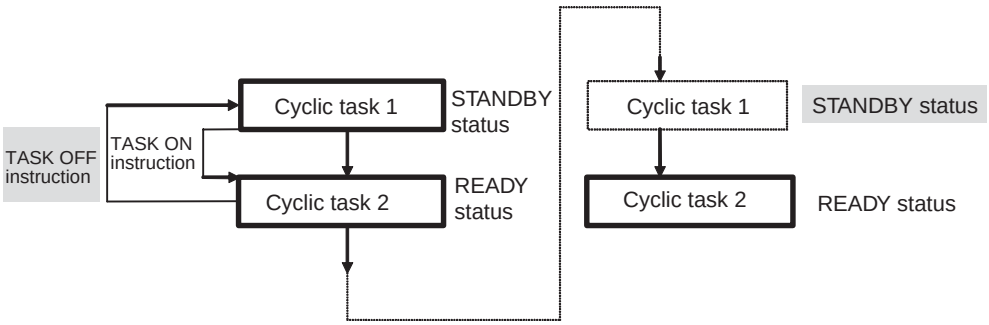


Tasks and the Execution Cycle

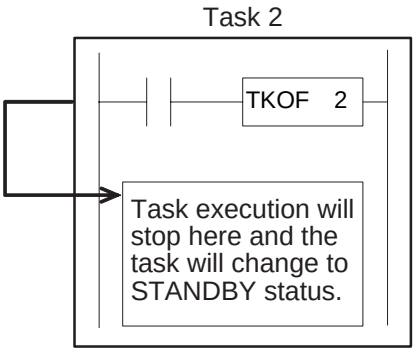
A cyclic task (including an extra cyclic task) that is in READY status will maintain that status in subsequent cycles.



A cyclic task that is in STANDBY status will maintain that status in subsequent cycles. The task will have to be turned ON using a TASK ON instruction in order to switch from STANDBY status to READY status.



If a TASK OFF instruction is executed for its own task, the task will stop being executed where the TASK OFF instruction is executed, and the task will change to STANDBY status.



Cyclic Task Numbers and the Execution Cycle

- If task m turns ON task n and m > n, task n will go to READY status at the next cycle.
  Example: If task 5 turns ON task 2, task 2 will go to READY status at the next cycle.
- If task m turns ON task n and m < n, task n will go to READY status in the same cycle.
  Example: If task 2 turns ON task 5, task 5 will go to READY status in the same cycle.
- If task m places task n in STANDBY status and m > n, will go to STANDBY status the next cycle.
  Example: If task 5 places task 2 in STANDBY status, task 2 will go to STANDBY status the next cycle.
- If task m places task n in STANDBY status and m < n, task n will go to STANDBY status in the same cycle.
  Example: If task 2 places task 5 in STANDBY status, task 5 will go to STANDBY status in the same cycle.

Relationship of Tasks to I/O Memory

- I/O memory other than Index Registers (IR) and Data Registers (DR) is shared by the tasks. For example, bit CIO 10.00 used in cyclic task 1 and bit CIO 10.00 used in cyclic task 2 refer to the same bit in memory. If I/O memory other than Index Registers (IR) and Data Registers (DR) accessed by more than one task, be sure to use sufficient caution when a value changed in one task is used in another task.
- There are two different ways to use Index Registers (IR) and Data Registers (DR): 1) Independently by task or 2) Shared by all task.
  - With independent registers, IR0 used by cyclic task 1 for example is different from IR0 used by cyclic task 2.
  - With shared registers, IR0 used by cyclic task 1 for example is the same as IR0 used by cyclic task 2.

The setting that determines if registers are independent or shared is made from the CX-Programmer.

I/O memory	Relationship to tasks
CIO, Auxiliary, Data Memory and all other memory areas except the IR and DR Areas.*1	Shared with other tasks.
Index registers (IR) and data registers (DR)*2	Used separately for each task.

*1 The current EM bank is also shared by tasks. Therefore if the current EM bank number is changed with cyclic task 1 for example, the new current EM bank number will be valid for cyclic task 2 as well.

*2 IR and DR values are not set when interrupt tasks are started. If IR and DR are used in an interrupt task, these values must be set by the MOVR/MOVRW (MOVE TO REGISTER and MOVE TIMER/COUNTER PV TO REGISTER) instructions within the interrupt task. After the interrupt task has been executed, IR and DR will return to their values prior to the interrupt automatically.

5-2-3 Interrupt Tasks

Interrupt tasks can be executed at any time in the cycle if any of the following conditions are in effect.

If an interrupt occurs, the interrupt task will be executed at any point in the cycle regardless of whether the CPU Unit is currently executing an instruction in a cyclic task, refreshing I/O, or performing peripheral servicing.

Types of Interrupt Tasks

The following types of interrupt tasks can be used.

- Power OFF Interrupt Task
  The power OFF interrupt task will be executed immediately before the power is turned OFF.
- Scheduled Interrupt Tasks
  A scheduled interrupt task is used in programs that require scheduling execution of certain parts of the user program, such as for monitoring the operation status at constant intervals.
- I/O Interrupt Tasks
  An I/O interrupt task will be executed when an input to an Interrupt Input Unit connected to the CPU Unit turns ON.
- External Interrupt Tasks
  An external interrupt task will be executed when an interrupt is requested by an Special I/O Unit or CPU Bus Unit.
- Extra Cyclic Tasks
  Extra cyclic tasks are interrupt tasks treated as cyclic tasks. In this case, the purpose is not interrupt processing, as is the case for other interrupt tasks, but to increase the number of cyclic tasks. Up to 128 cyclic tasks can be used. If extra cyclic tasks are used, however, up to 384 cyclic tasks can be used.

● List of Interrupt Tasks

Task	Task No.	Execution condition	Related settings
Power OFF Interrupt Task	1	Turing OFF the CPU Unit	• OFF interrupt tasks for PLC setup
Scheduled Interrupt Tasks 0 and 1*1	2 and 3	Each lapse of specific time on internal timer of CPU Unit	• Use the MSKS(690) (SET INTERRUPT MASK) instruction to set the interrupt interval. (1 to 9999) • See Scheduled Interrupt Time Units in PLC Setup. (10 ms/1.0 ms/0.1 ms)
I/O Interrupt Tasks 0 to 31*2	100 to 131	Input bit from an Interrupt Input Unit on the CPU Rack turning ON	• Use the MSKS(690) (SET INTERRUPT MASK) instruction to assign inputs from Interrupt Input Units on the CPU Rack.
External Interrupts 0 to 255*3	0 to 255	When requested by a Special I/O Unit or CPU Bus Unit on the CPU Rack*3	None (always valid)
Extra Cyclic Tasks 0 to 255	0 to 255	Every cycle while in the task is in READY status. (Task Control Instructions must be used.)	None (always valid)

*1 The scheduled interrupt tasks cannot be used if synchronous unit operation is being used.

*2 The Interrupt Input Unit must be connected in the CPU Rack. I/O Interrupt Units connected elsewhere cannot be used to request execution of I/O interrupt tasks.

*3 The Special I/O Unit or CPU Bus Unit must be connected in the CPU Rack. Units connected elsewhere cannot be used to generate external interrupts.

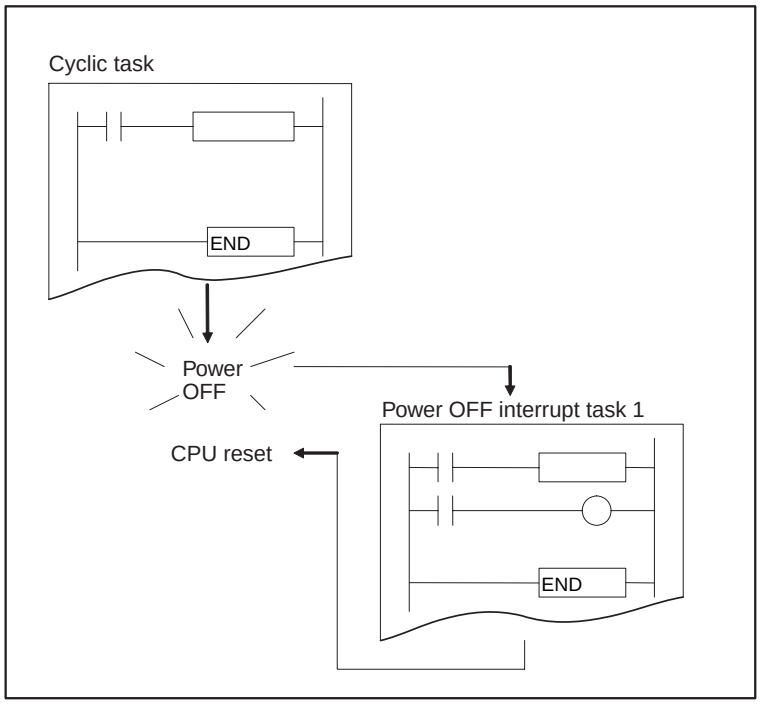
Precautions for Correct Use

Do not use SFC programs in interrupt tasks. They will not execute normally. SFC programs can be used, however, if the interrupt tasks is executed as a extra cyclic task.

ST programs can be used in interrupt tasks.

Power OFF Interrupt Task

This task is executed when the power supply is interrupted. When the power supply is interrupted, the Power Supply Unit will continue supplying 5 V of power to the CPU Unit for 10 ms, and the power OFF interrupt tasks will be executed during that time. (If a CJ1W-PD022 Power Supply Unit is used, the power will be supplied for only 1 ms, and so a power OFF interrupt task cannot be used.)



Interrupt Overhead Time for Power OFF Interrupt Task

The power OFF interrupt task is executed within 0.1 ms of the power being confirmed as being interrupted.

Settings for Executing Power OFF Interrupt Task

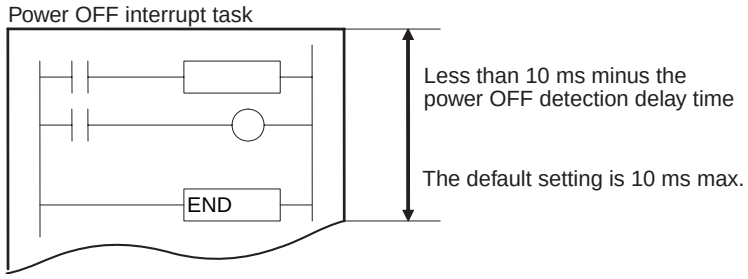
Enable the power OFF interrupt task on the Timings/Synchronous Tab Page in the PLC Settings Dialog Box of the CX-Programmer.

The power OFF interrupt task is disabled in the default PLC Setup at the start of cyclic task execution. To enable the power OFF interrupt task, clear the selection of the Power Off Interrupt disabled Option in the PLC Setup.

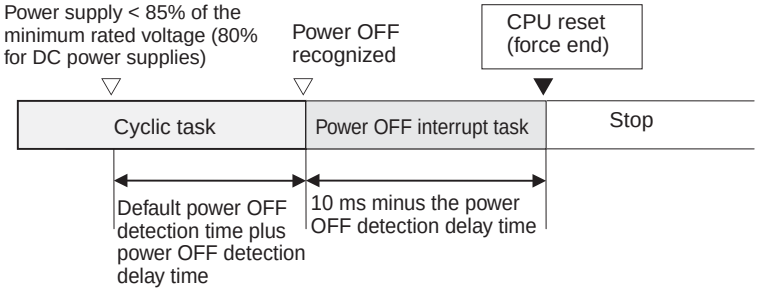
In the default PLC Setup, the power OFF interrupt task will be stopped after 10 ms. The power OFF interrupt task must be executed in less than 10 ms.

If a power OFF detection delay time is set in the PLC Setup, the power OFF interrupt task will be stopped after 10 ms minus the power OFF detection delay time setting in the PLC Setup. In this case, the power OFF interrupt task must execute in less than 10 ms minus the power OFF detection delay time set in the PLC Setup.

Example: If the power OFF detection delay time is set to 4 ms in PLC Setup, then execution time must be less than 10 minus 4 ms, or 6 ms.



Note A power OFF condition is recognized when the power supply falls below 85% of the minimum rated voltage (80% for DC power supplies), and the time it takes before the power OFF interrupt task actually executes is the default power OFF detection time (10 to 25 ms for AC power supplies and 2 to 5 ms for DC power supplies) plus the power OFF detection delay time in the PLC Setup (0 to 10 ms). Cyclic tasks will be executed for this amount of time.



● Restrictions on Using the Power OFF Interrupt Task

Execution Not Possible during Online Editing

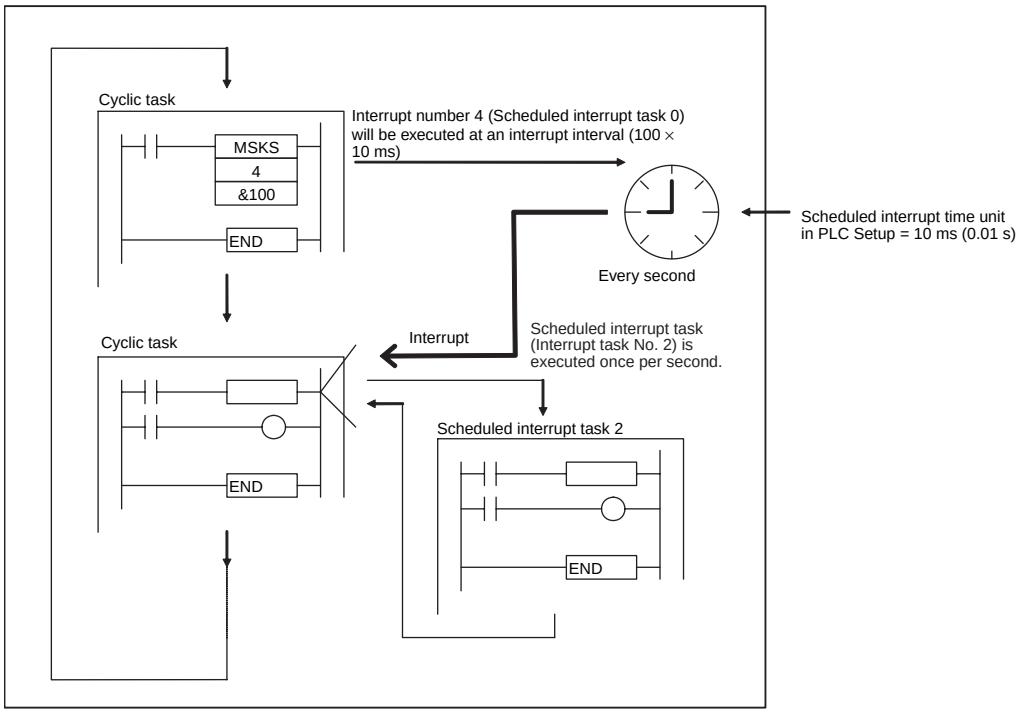
The power OFF interrupt task will not be executed if power is interrupted during online editing.

Instructions That Cannot Be Used in the Power OFF Interrupt Task

Some instructions cannot be used in the power OFF interrupt task, in addition to the instructions that cannot be used in any interrupt tasks. For details, refer to *5-2-4 Designing Tasks*.

Scheduled Interrupt Tasks

Tasks are executed at specified time intervals by using the timer in the CPU Unit. Up to two interrupt tasks (interrupt tasks 2 and 3) can be used for scheduled interrupts 0 and 1.



Interrupt Task Numbers and Interrupt Numbers

	Interrupt task No.	Interrupt number (MSKS(690))
Scheduled interrupt 0	2	4
Scheduled interrupt 1	3	5

Executing Scheduled Interrupt Tasks

Scheduled interrupt tasks are disabled when the CPU Unit operation is started. They can be enabled by using the MSKS(690) (SET INTERRUPT MASK) instruction. The time interval for scheduled interrupt tasks is determined by setting of the scheduled interrupt time unit in the PLC Setup and setting for the scheduled interrupt time specified with the operand when MSKS(690) is executed.

- Setting the Scheduled Interrupt Time Unit
  Set the unit for the interrupt time that is set with MSKS(690) using the “Scheduled Interrupt Interval” setting on the **Timings/Synchronous** Tab Page in the PLC Settings Dialog Box of the CX-Programmer.

  The unit can be set to 10 ms, 1.0 ms, or 0.1 ms. The default is 10 ms.

- Setting the Scheduled Interrupt Time Using MSKS(690)
  Set the interval at which the scheduled interrupt task will be executed.

  The unit can be set to between &1 and &9999 decimal (#0001 to #270F hex).

Example: If the scheduled interrupt time unit is set to 0.1 ms and the scheduled interrupt time is set to &5 decimal, the time interval is 0.1 x 5 = 0.5 ms.

The minimum time intervals that can be set for scheduled interrupt tasks are as follows:

- CJ2H CPU Units: 0.2 ms
- CJ2M CPU Units: 0.4 ms

However, a high-speed interrupt function can be used with CJ2H CPU Units with unit version 1.1 or later to set an interrupt interval of 0.1 ms for scheduled interrupt 0 (interrupt task 2). This setting cannot be used for other interrupts. For details on the high-speed interrupts, refer to *10-2-6 High-speed Interrupt Function*.

Precautions for Correct Use

- If you shorten the time interval for scheduled interrupts and increase the execution frequency for scheduled interrupt tasks, be careful because the time until execution of cyclic tasks and extra cyclic tasks is completed and the overall cycle time will increase.
- The scheduled interrupt tasks cannot be used if synchronous unit operation is being used.

● Interrupt Overhead Time for Scheduled Interrupt Tasks

The time from when the specified time set using MSKS(690) elapses until the interrupt tasks is actually executed is called the scheduled interrupt task startup time. The time it takes to return to the processing that was interrupted after the interrupt task program has been executed is called the cyclic task return time. The combination of the interrupt task startup time and the cyclic task return time is called the interrupt overhead time.

	Item	Time	
		CJ2H CPU Units	CJ2M CPU Units
Interrupt overhead time for scheduled interrupt tasks	Interrupt task startup time	22 μs or 13 μs*1 (27 μs for unit version 1.0)	30 μs
	Cyclic task return time	11 μs or 8 μs*1 (15 μs for unit version 1.0)	11 μs

*1 Using High-speed interrupt function
Refer to *10-2-6 High-speed Interrupt Function* for information on High-speed interrupt function.

● Resetting and Restarting with MSKS(690)

With CJ2M CPU Units, you can specify resetting the internal timer when you start a scheduled interrupt with MSKS(690) (a "reset start"). This enables creating a consistent time until starting the first interrupt without using CLI(691). When starting a scheduled interrupt, the scheduled interrupt time (i.e., the interval between one interrupt and the next) is set using MSKS(690). However, the time until the first time the scheduled interrupt task is started after MSKS(690) is executed depends on the present value of the internal timer. Therefore, the time to the first interval would be inconsistent if CLI(691) is not executed as well. For the CJ2M CPU Units, however, the internal timer can be reset when starting, making the time to the first interrupt consistent even if CLI(691) is not executed.

MSKS(690) Operand for Scheduled Interrupts

Operand	Set value
N (interrupt identifier)	4: Scheduled interrupt 0, normal operation (internal timer not reset, interrupt task 2)
	5: Scheduled interrupt 1, normal operation (internal timer not reset, interrupt task 3)
	14: Scheduled interrupt 0, reset start (CJ2M CPU Units only, interrupt task 2)
	15: Scheduled interrupt 1, reset start (CJ2M CPU Units only, interrupt task 3)

● Reading the PV of Internal Timers with MSKR(692)

With the CJ2M CPU Units, the present value of the internal timer used to measure the scheduled interrupt time can be read. Either the time that has elapsed until the scheduled interrupt is started or the time since the previous scheduled interrupt can be read. MSKR(692) is used to read the present value of the internal timer. The time unit is the same as the unit used for the scheduled interrupt time, i.e., the Scheduled Interrupt Interval set in the PLC Setup.

MSKR(692) Operand for Scheduled Interrupts

Operand	Set value
N (interrupt identifier)	4: Read scheduled interrupt time (set value) for interrupt task 0 (interrupt task 2) 5: Read scheduled interrupt time (set value) for interrupt task 1 (interrupt task 3) 14: Read present value of internal timer for interrupt task 0 (CJ2M CPU Unit only, interrupt task 2) 15: Read present value of internal timer for interrupt task 1 (CJ2M CPU Unit only, interrupt task 3)

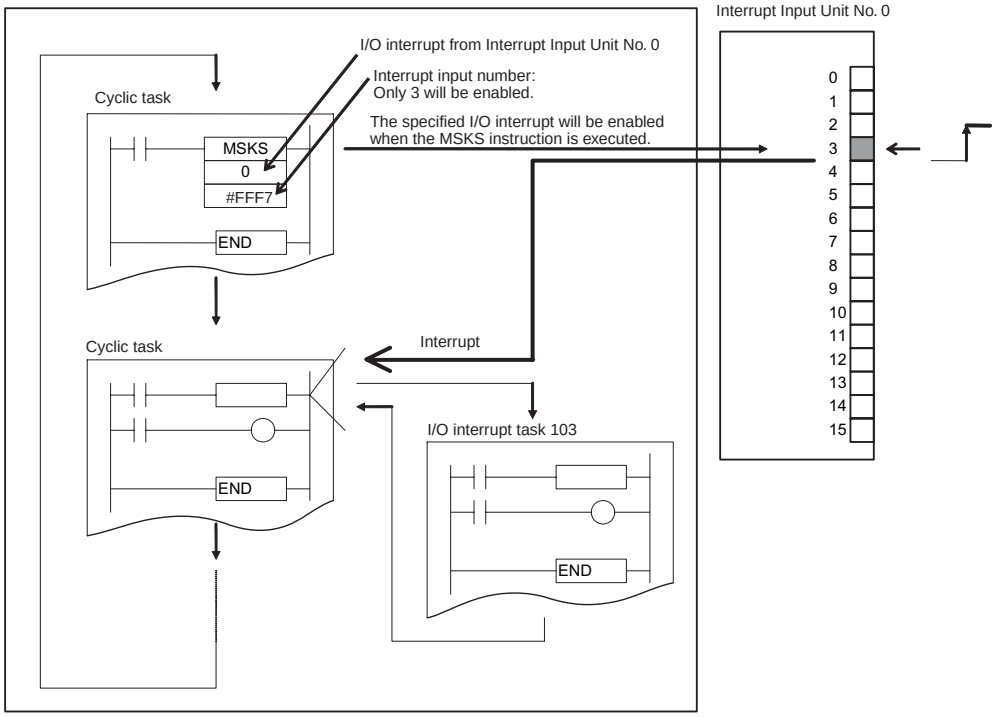
I/O Interrupt Tasks

An I/O interrupt task will be executed when an input to an Interrupt Input Unit turns ON. The maximum number of tasks that you can create is 32 (interrupt task numbers 100 to 131).

● Executing Interrupt Tasks

To enable I/O interrupts, execute the MSKS(690) (SET INTERRUPT MASK) instruction.

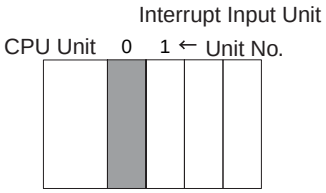
Example: The following example shows execution I/O interrupt task 103 when interrupt input No. 3 of Interrupt Input Unit No. 0 (the leftmost of the two Units 0 and 1) is ON.



Interrupt Input Unit Numbers, Input Numbers, and I/O Interrupt Task Numbers

Interrupt Input Unit No.*1	Input No.	I/O interrupt task
0	0 to 15	100 to 115
1	0 to 15	116 to 131

*1 Interrupt Input Unit numbers are in order from 0 to 1 starting from the CPU Unit.

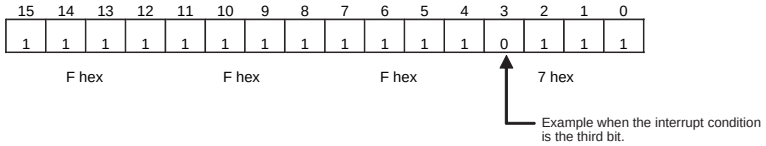


MSKS(690)

To enable I/O interrupt tasks, execute an MSKS(690) (SET INTERRUPT MASK) instruction in the program of a cyclic task, and enable the interrupt for the specified interrupt number of the specified Interrupt Unit.

Operand S (the Second Operand) of MSKS(690):
The bits of FFF7 hex correspond to the interrupt inputs of the Interrupt Input Unit. Interrupt input numbers 0 to 15 correspond to bits 0 to 15.

Example:



Precautions for Correct Use

Do not enable unneeded I/O interrupt tasks. If the interrupt input is triggered by noise and there is not a corresponding interrupt task, a fatal error (task error) will cause the program to stop.

Interrupt Overhead Time for I/O Interrupt Tasks

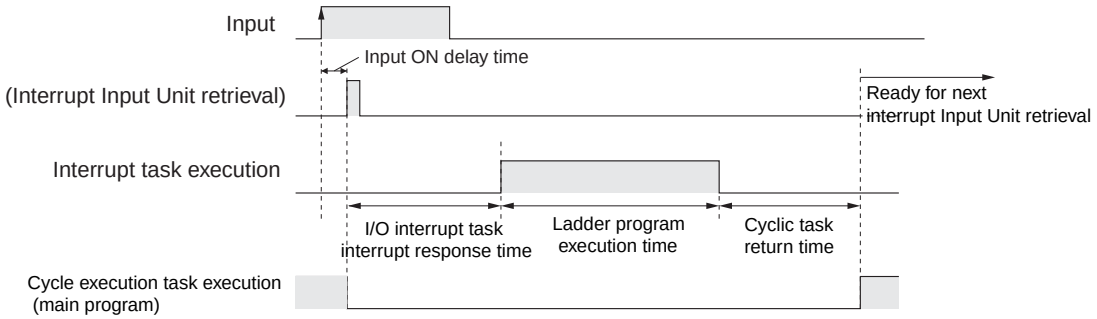
The time from when the input bit of the Interrupt Input Unit (CJ1W-INT01) turns ON (or OFF) until the CPU Unit receives the interrupt signal depends on the hardware response time. The time from when the interrupt signal is received until the I/O interrupt task is actually executed is called the interrupt task startup time in I/O interrupt tasks. The time it takes to return to the processing before interruption after the interrupt task program has been executed is called the cyclic task return time. The combination of the interrupt task startup time and the cyclic task return time is called the interrupt overhead time.

Using an Interrupt Input Unit

Item		Time	
		CJ2H CPU Units	CJ2M CPU Units
Hardware response (CJ1W-INT01)		Upward differentiation: 0.05 ms, Downward differentiation: 0.5 ms	
Interrupt overhead time	Interrupt task startup time	26 μs or 17 μs*1 (30 μs for unit version 1.0)	31 μs
	Cyclic task return time	11 μs or 8 μs*1 (15 μs for unit version 1.0)	10 μs

*1 Using High-speed interrupt function
Refer to *10-2-6 High-speed Interrupt Function* for information on High-speed interrupt function.

Note I/O interrupt tasks can be executed during user program execution, I/O refreshing, peripheral servicing, or overhead processing. (An I/O interrupt task can also be executed even if an instruction is being executed. The instruction will be interrupted.) An input interrupt will not be processed immediately if it occurs during execution of an interrupt task. The current interrupt task will be executed to the end first, and then execution of the new interrupt will be started after the cyclic task return time and interrupt task startup time have expired.

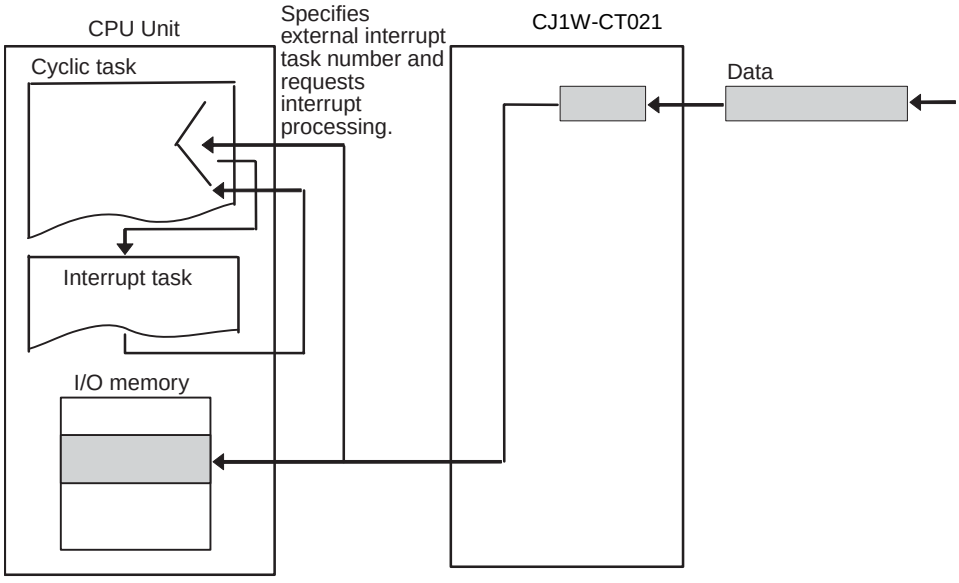


External Interrupts

Interrupt processing is performed at the CPU Unit in PLCs containing Special I/O Units or CPU Bus Units. Settings do not have to be made at the CPU Unit unless the program contains an external interrupt task for a particular task number. The Special I/O Unit or CPU Bus Unit must be connected in the CPU Rack to enable external interrupts.

Receiving external interrupt tasks is always enabled. The maximum number of tasks is 256 (interrupt task numbers 0 to 255).

Example: External Interrupt from a CJ1W-CT021 High-speed Counter Unit



To communicate to the CPU Unit the status information for the two external control inputs and 32 outputs of this Unit for other than I/O refreshing, set the external control inputs or outputs so that they trigger the external interrupt task.

Precautions for Correct Use

Do not use the same number for more than one interrupt task.

If an external interrupt task (0 to 255) has the same number as a power OFF task (task 1), scheduled interrupt task (task 2 or 3), or I/O interrupt task (100 to 131), the interrupt task will be executed for either interrupt condition (external interrupt or the other interrupt condition).

● Overhead Time for External Interrupt Tasks

For external interrupt tasks, the time until the CPU Unit receives the interrupt signal depends on the type of Unit or Board (i.e., Special I/O Unit or CJ-series CPU Bus Unit) that requests execution of the external interrupt task from the CPU Unit and the service type of the interrupt request. For details, refer to the relevant Unit manual.

The interrupt overhead time after the CPU Unit receives the interrupt signal is the same as for I/O interrupt tasks.

Extra Cyclic Tasks

An interrupt tasks can be executed every cycle, just like the normal cyclic tasks. Extra cyclic tasks (interrupt task numbers 0 to 255) are executed starting at the lowest task number after execution of the normal cyclic tasks (cyclic task numbers 0 to 127) has been completed. The maximum number of extra cyclic tasks is 256 (Interrupt task numbers: 0 to 255).

Precautions for Correct Use

Do not use task numbers assigned to extra cyclic tasks for interrupt tasks. An extra cyclic tasks with the same task number as a power OFF interrupt task, a scheduled interrupt task, or an I/O interrupt task will be executed both as an extra cyclic task and as an interrupt task.

Note 1 TKON(820) and TKOF(821) can be input and executed in an extra cyclic task, but they will not be executed when the task is executed as an interrupt task.

2 The differences between normal cyclic tasks and extra cyclic tasks are listed in the following table.

Item	Extra cyclic tasks	Normal cyclic tasks
Activating at startup	Setting is not possible.	Set from CX-Programmer
Task Flags	Not supported.	Supported. (Cyclic task numbers 0 to 127 correspond to Task Flags TK000 to TK127.)
Initial Task Execution Flag (A200.15) and Task Start Flag (A200.14)	Not supported.	Supported.
Index (IR) and data (DR) register values	Not defined when task is started (same as normal interrupt tasks). Values at the beginning of each cycle are undefined. Always set values before using them. Values set in the previous cycle cannot be read.	Undefined at the beginning of operation. Values set in the previous cycle can be read.

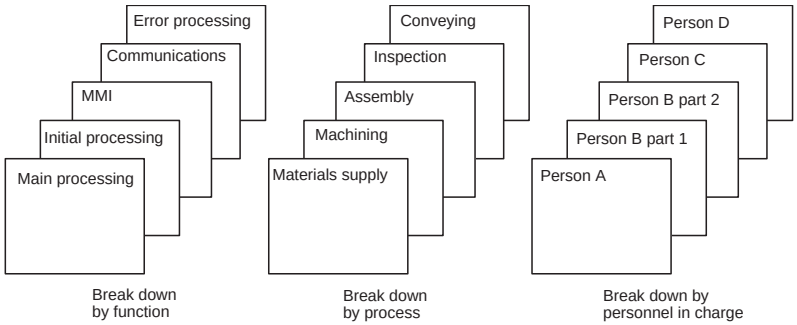
5-2-4 Designing Tasks

Guidelines

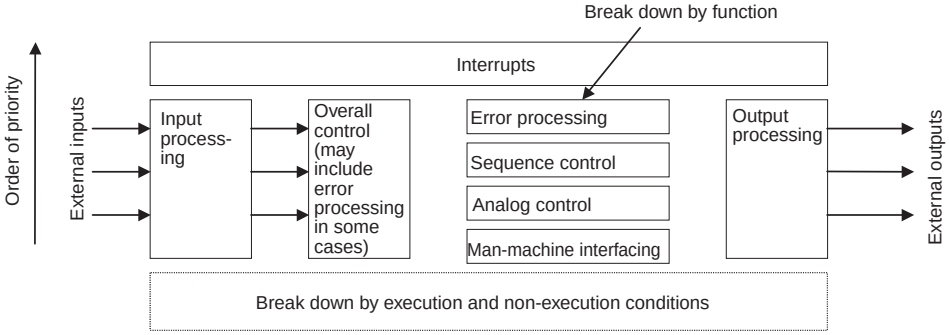
Task design is important to build a system with a high degree of reliability and easy maintenance. Pay attention to the following points.

- Divide programming into tasks based on the following.
  - Consider specific conditions for execution and non-execution with an understanding of status transitions.
  - Consider the presence or absence of external I/O.
  - Consider functions and personnel in charge.
  - Consider execution in order of priority. Separate processing into cyclic and interrupt tasks. Assign the lowest number to the task with the highest priority.

Example 1: Designing Tasks by Function and Personnel in Charge.

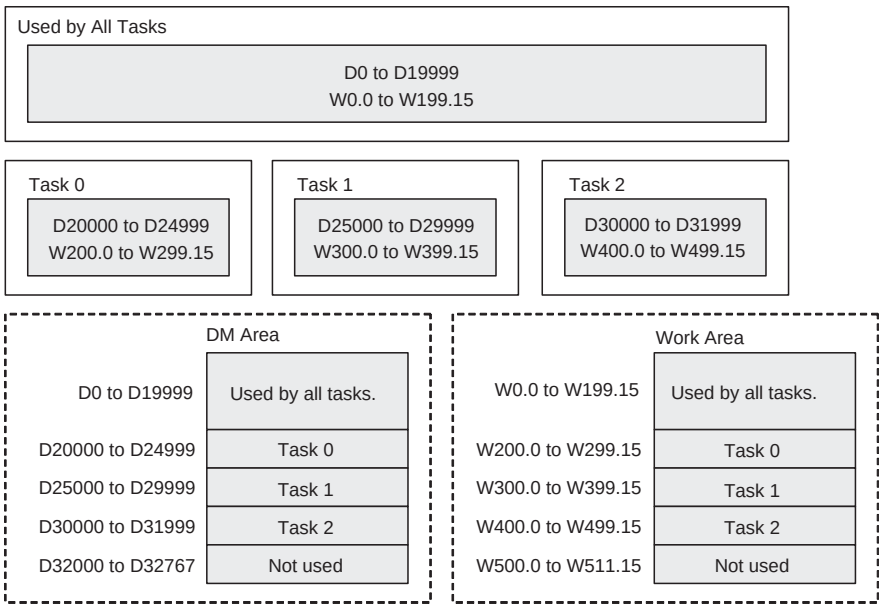


Example 2: Designing Tasks by Execution Priority
Allocate lower numbers to control tasks than to processing tasks.



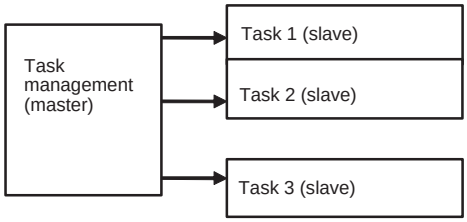
- Break down and design programs to ensure autonomy. Keep the amount of data exchanged between tasks to an absolute minimum.

  Organize the I/O memory used by all tasks and the I/O memory used only in individual tasks. Organize and allocate a range of I/O memory for use only by each task.

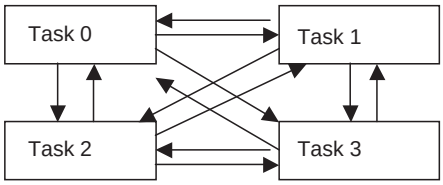


- Use one task to start and stop other tasks and thus control the overall user program.

  Good Example: Define a master-slave relationship between tasks.



Bad Example: Control is difficult if starting and stopping tasks is performed in a disorganized manner.



- Allocate lower numbers to high-priority interrupt tasks.
- A task in READY status will be executed in subsequent cycles as long as the task itself or another task does not change it to STANDBY status. Be sure to insert a TKOF(821) (TASK OFF) instruction for other tasks if processing is to be branched between tasks.
- Use the Initial Task Execution Flag (A200.15) or the Task Start Flag (A200.14) in the execution condition to execution instructions to initialize tasks. The Initial Task Execution Flag will be ON during the first execution of each task. The Task Start Flag each time a task enters READY status.

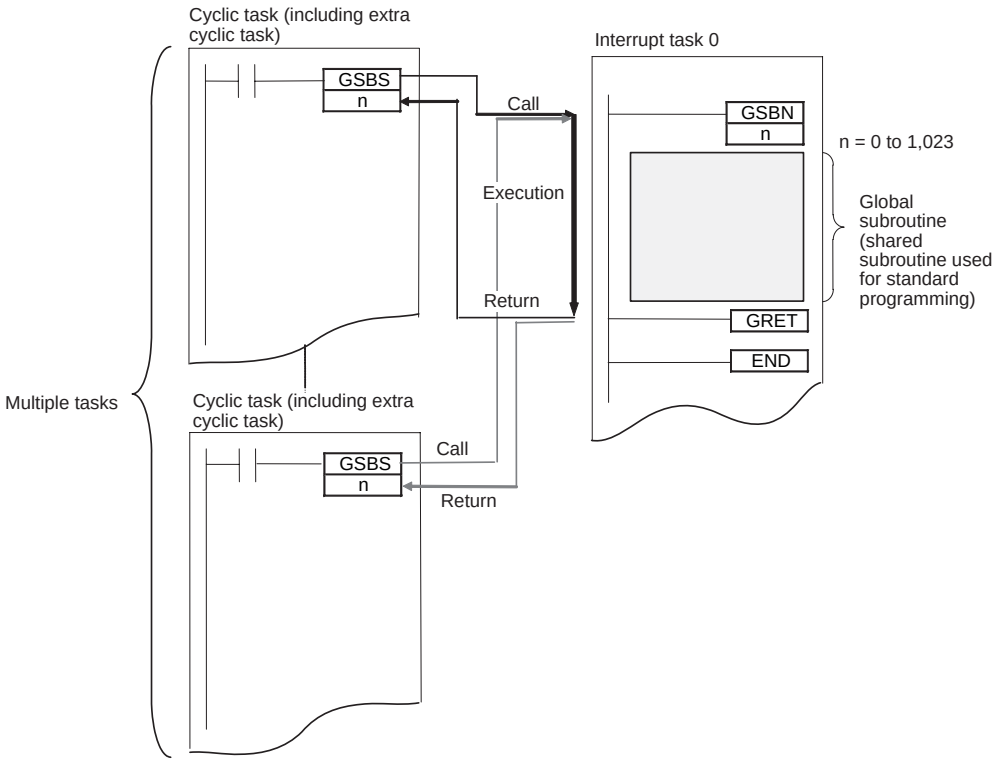
Additional Information

Global Subroutines

With regular subroutine instructions, it is not possible to call a subroutine in one task from a different task. Global subroutines can be created in interrupt task number 0, and these subroutines can be called from any cyclic task (including extra cyclic tasks).

The GSBS instruction is used to call a global subroutine. The subroutine number must be between 0 and 1,023. The global subroutine is defined at the end of interrupt task number 0 (just before END(001)) between the GSBN and GRET instructions.

Global subroutines can be used to create a library of standard program sections that can be called whenever necessary.



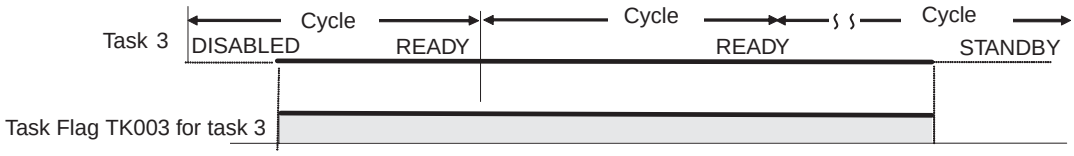
Flags Related to Tasks

● Flags Related to Cyclic Tasks

The following flag work only for normal cyclic tasks. They do not work for extra cyclic tasks.

Task Flags (TK000 to TK127)

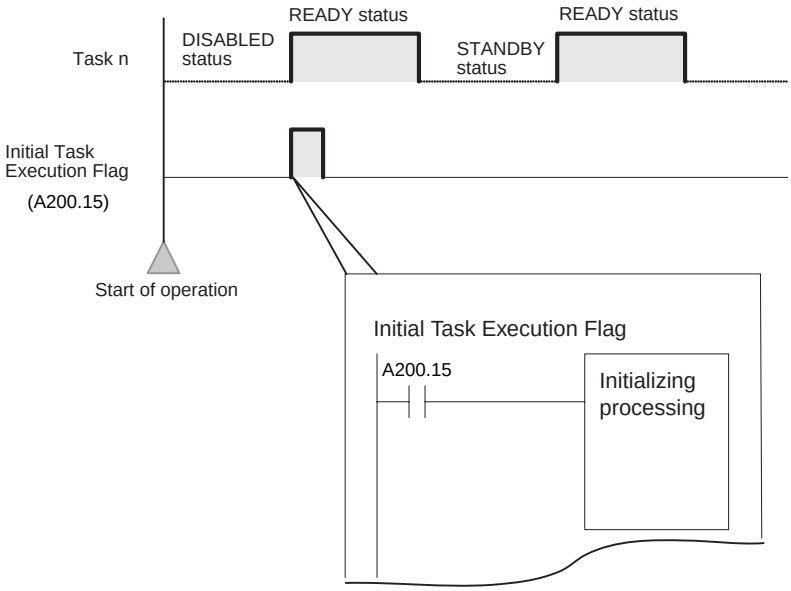
Use this flag to check if the task is being presently being executed. A Task Flag is turned ON when a cyclic task is in READY status and is turned OFF when the task is in DISABLED (INI) or in STANDBY (WAIT) status. Task numbers 00 to 127 correspond to TK000 to TK127.



Note Task Flags are used only with cyclic tasks and not with interrupt tasks. For interrupt tasks, use flags related to the interrupt tasks, such as word A440 (Maximum Interrupt Task Processing Time) or word A441 (Interrupt Task with Maximum Processing Time). Refer below for information on flags related to interrupt tasks.

Initial Task Execution Flag (A200.15)

Use this flag to perform initial processing only once during operation. It will turn ON the first time a task is executed and will turn OFF when execution of the task has been completed.

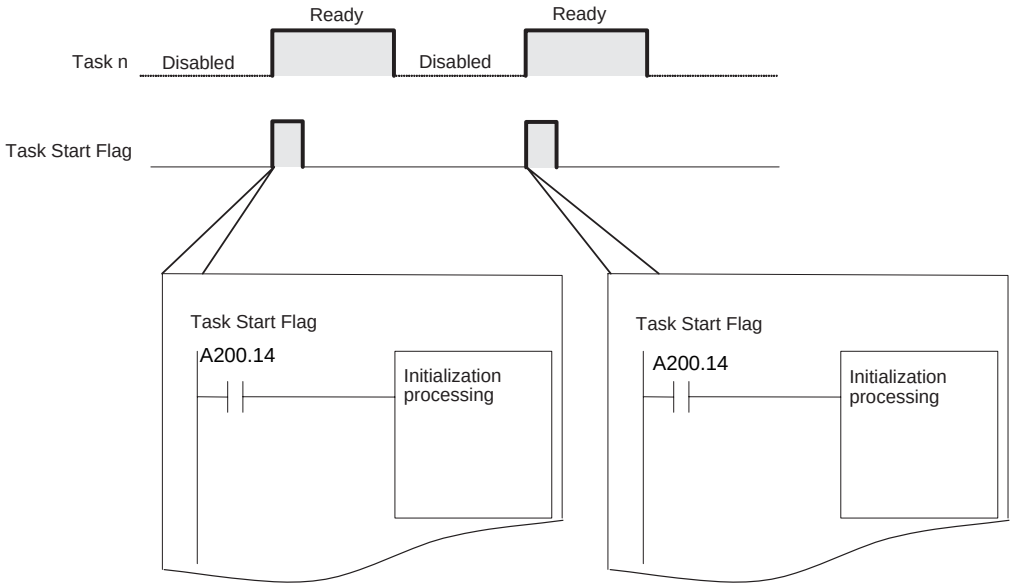


The Initial Task Execution Flag will turn ON when a task is executed the first time. Use this flag to trigger initializing processing.

Note Even though a STANDBY cyclic task is changed back to READY status using the TKON(820) instruction, this is not considered an initial execution, and the Initial Task Execution Flag (20015) will not turn ON again. Use the Task Start Flag (A200.14).

Task Start Flag (A200.14)

The Task Start Flag turns ON whenever cyclic task status changes from STANDBY status to READY status (whereas the Initial Task Execution Flag (A200.15) turns ON only once, the Task Start Flag will turn ON every time the status changes from STANDBY status to READY status).



The Task Start Flag can be used to perform initialization processing whenever a task goes from STANDBY to READY status, i.e., when a task on STANDBY is enabled using the TKON(820) instruction.

● Interrupt Task Flags and Words

Maximum Interrupt Task Processing Time (A440)*

The maximum processing time for an interrupt task is stored in binary data in 0.1-ms units and is cleared at the start of operation.

Interrupt Task with Maximum Processing Time (A441)*

The interrupt task number with maximum processing time is stored in binary data. Here, 8000 to 80FF hex correspond to task numbers 00 to FF Hex.

This data is cleared at the start of operation.

Duplicate Refresh Error Flag (Nonfatal Error) (A402.13)

If Duplicate Refresh Error Detection is enabled in the PLC Setup, the Duplicate Refresh Error Flag will turn ON if a duplicate refresh error occurs.

* The values in words A440 and A441 will not be valid if High-speed interrupt function is enabled in the PLC Setup.

Duplicate Refresh Error Cause Flag (A426.15)/Task Number Generating the Duplicate Refresh Error (A426.00 to 426.11)

A402.13	Interrupt Task Error Description	A426.15	A426.00 to 426.11
Interrupt Task Error (when Interrupt Task Error Detection is enabled in the PLC Setup)	If Duplicate Refresh Error Detection is enabled in the PLC Setup, the Duplicate Refresh Error Flag will turn ON if the following conditions occur for the same Special I/O Unit. • There is a conflict between an IORF, FIORF, IORD, or IOWR instruction executed in the interrupt task and an IORF, FIORF, IORD, or IOWR instruction executed in the cyclic task. • There is a conflict between an IORF, FIORF, IORD, or IOWR instruction executed in the interrupt task and the CPU Unit's I/O refreshing (END refreshing). When a Special I/O Unit's Cyclic Refreshing is enabled in the PLC Setup, and an IORF, FIORF, IORD, or IOWR instruction is executed for the same Special I/O Unit, there will be duplicate refreshing and an Interrupt Task Error will occur.	1 (ON)	The unit number of the Special I/O Unit being refreshed will be stored in 12 bits of binary data (unit No. 0 to 95: 000 to 05F Hex).

Task Number when Program Stopped (A294)

The type of task and the current task number when a task stops execution due to a program error will be stored as follows:

Type	A294
Interrupt task	8000 to 80FF Hex (correspond to interrupt task numbers 0 to 255)
Cyclic task	0000 to 007F Hex (correspond to task numbers 0 to 127)

● Flags Related to All Tasks

Task Error Flag (A295.12)

The Task Error Flag will turn ON if one of the following task errors occurs.

- No cyclic tasks are READY during a cycle.
- The program allocated to a cyclic task (including extra cyclic tasks) does not exist. (This situation will not occur when using the CX-Programmer.)
- No program is allocated to an activated interrupt task (including extra cyclic tasks).

Task Number When Program Stopped (A294)

The type of task and the current task number when a task stops execution due to a program error will be stored as follows:

Type	A294
Cyclic task	0000 to 007F Hex (correspond to task numbers 0 to 127)
Interrupt task (including extra cyclic task)	8000 to 80FF Hex (correspond to interrupt task numbers 0 to 255)

This information makes it easier to determine where the fatal error occurred, and it will be cleared when the fatal error is cleared. The program address where task operation stopped is stored in A298 (rightmost bits of the program address) and in A299 (leftmost bits of the program address).

Precautions

● Instruction Usage Restrictions

Instructions That Must Be in the Same Task

The following instructions must be placed in the same task.

Mnemonic	Instruction
JMP/JME	JUMP/JUMP END
CJP/JME	CONDITIONAL JUMP/JUMP END
CJPN/JME	CONDITIONAL JUMP NOT/CONDITIONAL JUMP END
JMP0/JME0	MULTIPLE JUMP/JUMP END
FOR/NEXT	FOR/NEXT
IL/ILC	INTERLOCK/INTERLOCK CLEAR
SBS/SBN/RET	SUBROUTINE CALL/SUBROUTINE ENTRY/SUBROUTINE RETURN (If the global subroutine instructions (GSBS(750), GSBN(751), and GRET(752)) are used, the subroutine can also be called from another task.)
MCRO/SBN/RET	MACRO/SUBROUTINE ENTRY/SUBROUTINE RETURN
BPRG/BEND	BLOCK PROGRAM BEGIN/BLOCK PROGRAM END
STEP/STEP	STEP DEFINE

Instructions That Are Not Allowed in Interrupt Tasks

The following instructions cannot be placed in interrupt tasks. Any attempt to execute one of these instructions in an interrupt task will cause the Error Flag (P_ER) to turn ON and the instruction will not be executed. The following instructions can be used if an interrupt task is being used as an extra cyclic task.

Mnemonic	Instruction
TKON	TASK ON
TKOF	TASK OFF
STEP	STEP DEFINE
SNXT	STEP NEXT
STUP	CHANGE SERIAL PORT SETUP
DI	DISABLE INTERRUPT
EI	ENABLE INTERRUPT
SFCON, SFCOFF, SFCPR, SFCPRN	SFC Task Control Instructions

- The operation of the following instructions is unpredictable in an interrupt task: HUNDRED-MS TIMER: TIM and TIMX(550), TEN-MS TIMER: TIMH(015) and TIMHX(551), ONE-MS TIMER: TMHH(540) and TMHHX(552), TENTH-MS TIMER: TIMU(541) and TIMUX(556), HUNDREDTH-MS TIMER TIMUH(544) and TIMUHX(557), ACCUMULATIVE TIMER: TTIM(087) and TTIMX(555), MULTIPLE OUTPUT TIMER: MTIM(543) and MTIMX(554), LONG TIMER: TIML(542) and TIMLX(553), TIMER WAIT: TIMW(813) and TIMWX(816), HIGH-SPEED TIMER WAIT: TMHW(815) and TMHWX(817), PID CONTROL: PID(190), and FAILURE POINT DETECTION: FPD(269).
- In addition to the above, the following instruction cannot be used in the power OFF interrupt task. The Error Flag (P_ER) will not turn ON if these instructions are used in the power OFF interrupt task. The instruction will not be executed.
  READ DATA FILE: FREAD(700), WRITE DATA FILE: FWRIT(701), WRITE TEXT FILE(TWRIT(704)), NETWORK SEND: SEND(090), NETWORK RECEIVE: RECV(098),

DELIVER COMMAND: CMND(490), TRANSMIT: TXD(236), RECEIVE: RXD(235), and PROTOCOL MACRO: PMCR(260), EXPLICIT MESSAGE SEND: EXPLT(270), EXPLICIT GET ATTRIBUTE: EGATR(271), EXPLICIT SET ATTRIBUTE: ESATR(272), EXPLICIT WORD READ: ECHRD(273), EXPLICIT WORD WRITE: ECHWR(274), TRANSMIT VIA SERIAL COMMUNICATIONS UNIT: TXDU(256), and RECEIVE VIA SERIAL COMMUNICATIONS UNIT: RXDU(255).

- The following instructions cannot be used in interrupt tasks when high-speed interrupts are enabled in the PLC Setup of a CJ2H CPU Unit. An error will occur if any of them are executed.

  SELECT EM BANK (EMBC(281)), NETWORK SEND (SEND(090)), NETWORK RECEIVE (RECV(098)), DELIVER COMMAND (CMND(490)), PROTOCOL MACRO (PMCR(260)), TRANSMIT VIA SERIAL COMMUNICATIONS UNIT (TXDU(256)), RECEIVE VIA SERIAL COMMUNICATIONS UNIT (RXDU(255)), EXPLICIT MESSAGE SEND (EXPLT(720)), EXPLICIT GET ATTRIBUTE (EGATR(721)), EXPLICIT SET ATTRIBUTE (ESATR(722)), EXPLICIT WORD READ (ECHRD(723), and EXPLICIT WORD WRITE (ECHWR(724))

  An instruction processing error will also occur for table data processing instructions, character string processing instructions, or data shift instructions if background processing is specified in the PLC Setup.

Additional Information

Relationship of Tasks to Timer Operation

Timer instructions operate as follows:

- Timer Numbers 0 to 2047
  Timer present values will continue to be updated even if the task containing the timer is changed to STANDBY status or is returned to READY status. If the task containing an active TIM instruction goes to STANDBY status and the present value is 0 when the task returns to READY status, the Completion Flag will turn ON as soon as the TIM instruction is executed. (Completion Flags for timers are updated only when the instruction is executed.) The present value will continue to be updated.
- Timer Numbers 2048 to 4095
  The present values of timers that have been started in a task will not be updated while the task is in STANDBY status. There is a setting in the PLC properties, however, that can be used to enable the same operation for timers 2048 to 4095 as for timers 0 to 2047.

Relationship of Tasks to Condition Flags

All Condition Flags will be cleared before execution of each task. Therefore Condition Flag status at the end of task 1 cannot be read in task 2. CCS(282) and CCL(283) can be used to read Condition Flag status from another part of the program, e.g., from another task.

● Precautions on Interrupt Tasks

Interrupt Task Priority

Execution of another interrupt task will be stopped to allow the power OFF interrupt task to execute. The CPU Unit will be reset, but the stopped interrupt task will not be executed after the execution of the power OFF interrupt task.

The priority for other interrupt tasks is described in the following sections.

Multiple Interrupts Occurring Simultaneously

Interrupt tasks other than power OFF interrupt tasks will be executed in the following order of priority whenever multiple interrupts occur simultaneously.

I/O interrupt tasks > External interrupt tasks > Scheduled interrupt tasks

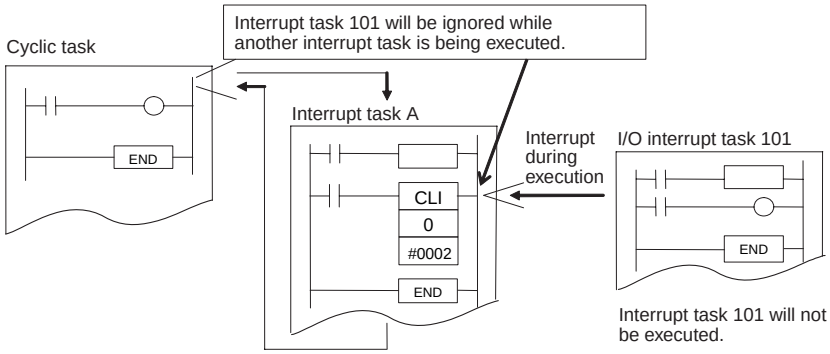
Each of the various types of interrupt task will be executed in order starting from the lowest number if more than one occurs.

Interrupt during Interrupt Task Execution

If an interrupt occurs while another interrupt task is being executed, the task for the interrupt will not be executed until the original interrupt finishes executing.

Additional Information

If you do not want a specific I/O interrupt task number to be saved and executed for the CPU Unit when it occurs while another interrupt task is being executed, execute the CLI (CLEAR INTERRUPT) instruction from the other interrupt task to CLEAR the interrupt number saved internally. Scheduled interrupts and external interrupts cannot be cancelled.



Additional Information

Only one interrupt will be recorded in memory for each interrupt task and an interrupt will not be recorded for an interrupt that is already being executed. Because of the low order of priority of scheduled interrupts and because that only one interrupt is recorded at a time, it is possible for a scheduled interrupt to be skipped.

● Application Precautions

Executing IORF(097), FIORF(225), IORD(222), or IOWR(223) for a Special I/O Unit

If a Special I/O Unit is being used and IORF(097), FIORF(225), IORD(222), or IOWR(223) will be executed from an interrupt task, always disable cyclic refreshing for that Special I/O Unit in the PLC Setup.

If a Special I/O Unit is being refreshed by cyclic refreshing or an I/O refreshing instruction, a duplicate refresh error will occur if you try to refresh the same Special I/O Unit with an IORF(097) or FIORF(225) instruction in an interrupt task or if an attempt is made to read/write data for the same Special I/O Unit with an IORD(222) or IOWR(223) instruction. In this case, the IORF(097), FIORF(225), IORD(222), or IOWR(223) instruction will not be executed, but the Error Flag (P_ER) will not be turned ON. Cyclic refreshing will be performed normally.

If Duplicate Refresh Error Detection is enabled in the PLC Setup when a duplicate refresh error occurs, A402.13 (Duplicate Refresh Error Flag) will turn ON and the unit number of the Special I/O Unit will be stored in A426 (Duplicate Refresh Error Task Number).

● Prohibiting Interrupts during Specific Processing with Cyclic Tasks

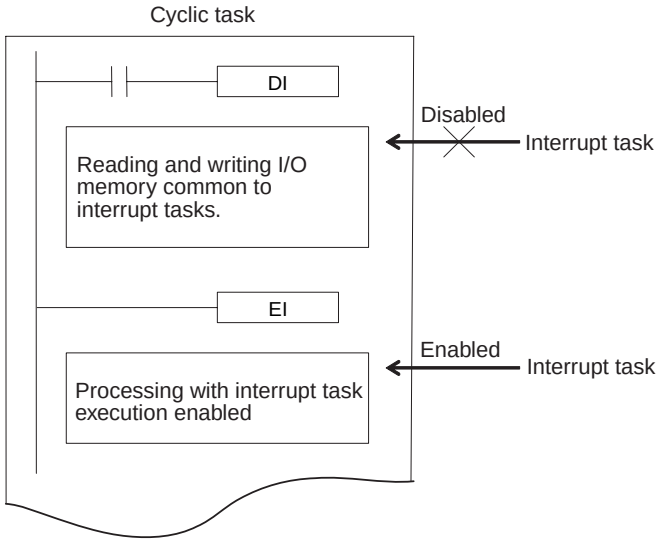
Processing will be interrupted and the interrupt task will be executed if an interrupt occurs during in the following.

- While an instruction is being executed
- During Basic I/O Unit or Special I/O Unit refreshing
- During peripheral servicing

Ensuring Data Concurrency between Cyclic and Interrupt Tasks

Data may not be concurrent if a cyclic task and an interrupt task are reading and writing the same I/O memory addresses. Use the following procedure to disable interrupts during memory access by cyclic task instructions.

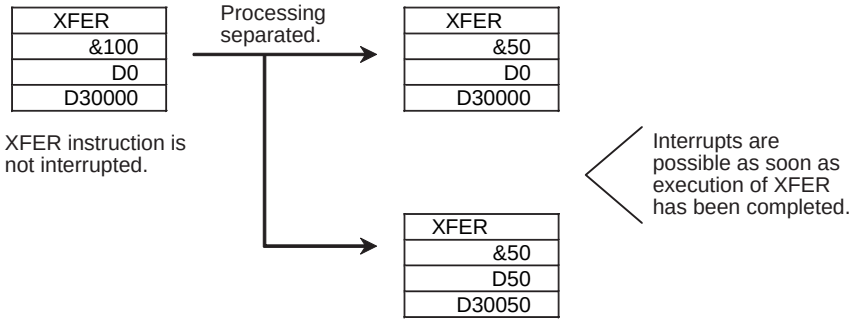
- Immediately prior to reading or writing by a cyclic task instruction, use a DI(693) (DISABLE INTERRUPT) instruction to disable execution of interrupt tasks.
- Use an EI(694) (ENABLE INTERRUPT) instruction immediately after processing in order to enable interrupt task execution.



Problems may occur with data concurrency even if DI(693) and EI(694) are used to disable interrupt tasks during execution of an instruction that requires response reception and processing (such as a network instruction or serial communications instruction).

Additional Information

Execution of the BIT COUNTER (BCNT), BLOCK SET (BSET), and BLOCK TRANSFER (XFER) instructions will not be interrupted for execution of interrupt task, i.e., execution of the instruction will be completed before the interrupt task is executed, delaying the response of the interrupt. To prevent this, separate data processing for these instructions into more than one instructions, as shown below for XFER.

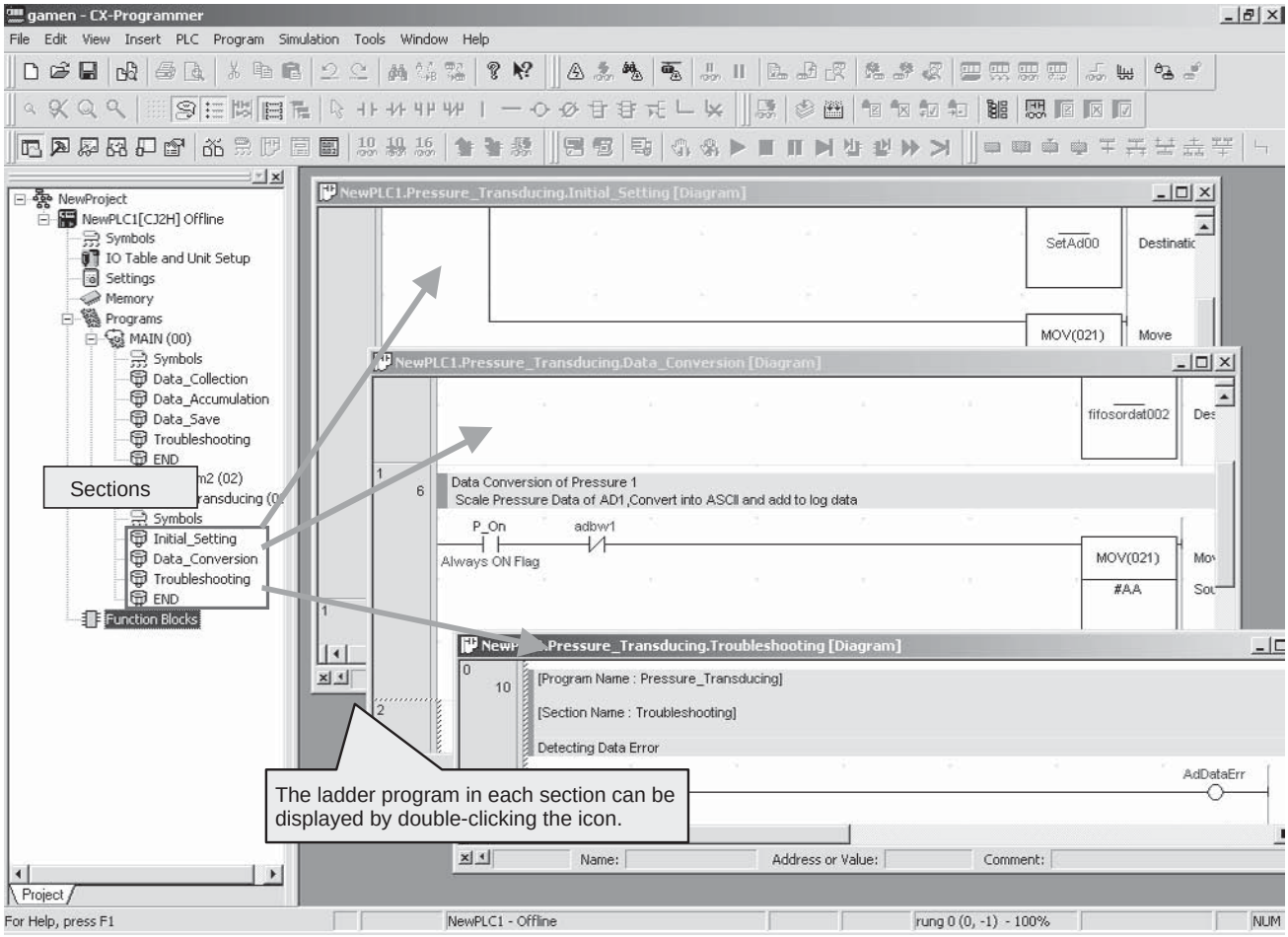


5-3 Sections

5-3-1 Overview of Sections

Programs can be created and displayed in sections with the CX-Programmer to divide programming into functional units. Any program in a task can be divided into sections. Using sections improves program legibility and simplifies editing.

- Clearer Processing
  A section can be given any name to indicate the process or control performed in that section
- Easy Editing
  The order of sections can be changed by dragging and dropping them with the mouse, and sections can be copied and pasted.
- Unlimited Use
  There is no limit to the number of sections that can be created per program.



● Number of Sections

There is no limit to the number of sections that you can use. Also, as previously, it is possible to created a program with only one section.

● Order of Section Execution

Section are executed in the order they appear in the project tree (i.e., top to bottom). Program addresses automatically continue consecutively in the order the tasks are executed. The rung number starts from 0 in each section.

Section Names

The section name will be “Section□” with serial numbers starting from 1 automatically allocated as the suffix. The names can be changed using user definitions.

Advantages of Dividing Programs into Sections

Dividing programming into sections offers the following advantages.

- Programming can be uploaded from the CPU Unit one section at a time. If one section of the programming is uploaded in advance, the time required to start online editing will be shortened. Only one section of the program can be downloaded at a time for online editing. It is also not possible to download sections one at a time.
- Sections can be moved to change the order of execution of the overall program, they can be moved to other programs, and they can be copied on the project tree.

This enables standardizing programming modules that are smaller than the overall program.

File Memory Files Related to Sections

Sections are included in the CX-Programmer project file (.CXP). The section names, section comments, and program comments for one CPU Unit are stored in the program index file (PROGRAMS.IDX).



Additional Information

Tasks are used to control whether task program are executed. On the other hand, sections are divisions created at a lower level in user-defined programs. Sections are created mainly to make programming easier to understand, as in the following application examples.

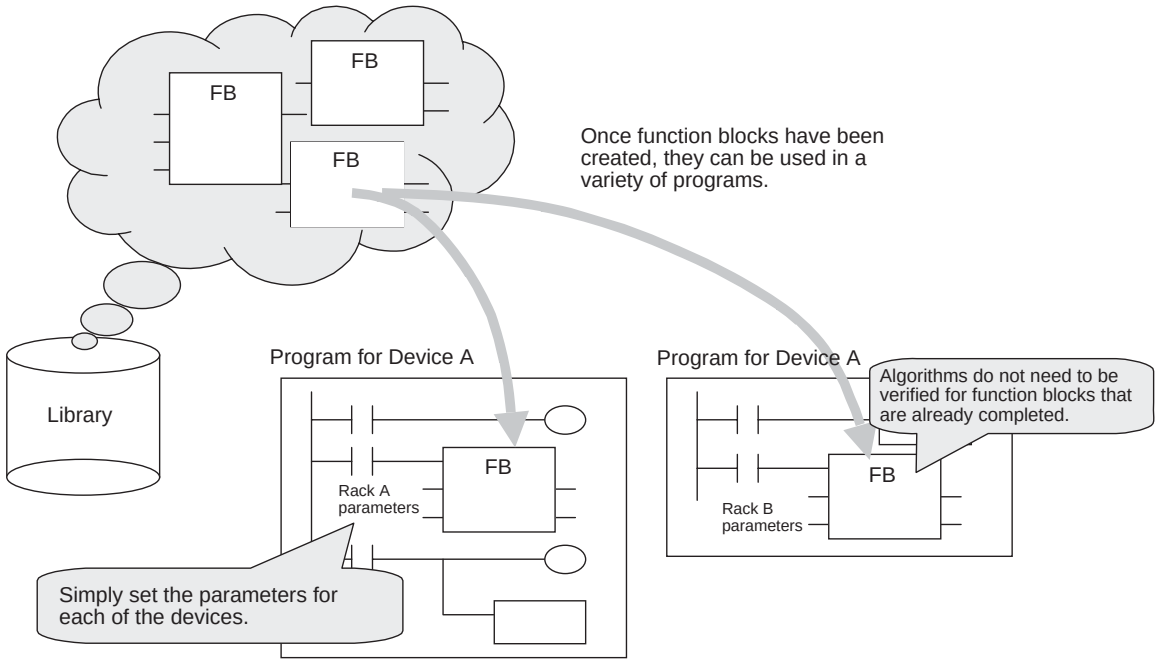
- Initial processing and main processing can be divided into different sections in one task when the same processing is required whenever the task is executed.
- The normal program area and subroutines can be divided into sections.

5-4 Function Blocks

5-4-1 Function Blocks

A function block (FB) is a basic program element containing the programming for a standard processing function that has been defined in advance. Once you have created function blocks and saved them in a library, you can use them in programming simply by selecting the function blocks that are required for the system, placing them in the program, and setting I/O parameters. Excellent function blocks can greatly improve the efficiency of program development.

OMRON function blocks conform to IEC 61131-3.



● Features of Function Blocks

- Reusability
  A function block can be saved in a library so that it can be easily reused. Once the programmer understands the function blocks, the programmer can use them simply by setting parameters. This greatly improves programming efficiency.
- Reduced Debugging Work
  Reusing high-quality function blocks that have already been debugged eliminates the need to debug them again and thereby reduces debugging work.
- Data Protection
  The variables inside a function block cannot be accessed directly from the outside, so the data is protected. (Data will not be changed unintentionally.)

5-4-2 Features of Function Blocks

● Entering Addresses for Function Blocks

Addresses in function blocks are generally entered by using variables rather than the actual physical addresses. The programmer uses these variables to set the parameters (i.e., addresses or values). The address used for each variable is automatically allocated by the CX-Programmer every time a function block is placed in the program.

● Nesting

A function block can be nested within another function block. Up to eight nesting levels are possible, including calling the first function block.

● Smart FB Library

The Smart FB Library is a set of function blocks that improve operation between OMRON PLC Units and FA components. It is not necessary to create a ladder program to use basic Unit and FA component functions.

● Online Editing of FB Definitions

FB definitions can be changed during operation, so FB definitions can be edited quickly during debugging. In addition, FBs can be used with confidence even in equipment that must operate 24 hours/day.

● Protecting FB Definitions

It is possible to prevent unauthorized manipulation, editing, or misappropriation of the program by setting passwords for the function block definitions allocated in the project file and protecting the definitions based on their purpose.

● FB Generation Function

Existing PLC programming can be reused by easily converting it to FBs.

5-4-3 Function Block Specifications

Item		Specifications	
Defining and creating function blocks	Number of function block definitions	CJ2H-CPU6□(-EIP) or CJ2M-CPU□5/□4: 2,048 max. CJ2M-CPU□3/□2/□1: 256 max.	
	Function block names	64 characters max.	
	Variables	Variable names	15,000 characters max.
		Variable types	Input variables (Inputs), output variables (Outputs), input-output variables (In Out), internal variables (Internals), and external variables (Externals)
		Number of variables used in a function block (not including internal variables, external variables, EN, and EN0)	Maximum number of variables per function block definition • Input-output variables: 16 max. • Input variables + input-output variables: 64 max. • Output variables + input-output variables: 64 max.
		Allocation of addresses used by variables	Automatic allocation (The allocation range can be set by the user.)
		Actual address specification	Supported
		Array specifications	Supported (one-dimensional arrays only and only for internal variables and input-output variables)
	Language	Function blocks can be created in ladder programming language or structured text.	
Creating instances	Number of instances	CJ2H-CPU6□(-EIP) or CJ2M-CPU□5/□4: 2,048 max. CJ2M-CPU□3/□2/□1: 256 max.	
	Instance names	15,000 characters max.	
FB Program Area		20K steps Supported only by the CJ2M CPU Units. This area is used to store function block definitions.	
Storing function blocks as files	Project files	The project file (.cxp/cxt) Includes function block definitions and instances.	
	Program/network symbol files	The file memory program file (*.obj) includes function block definitions and instances.	
	Function block library files	Each function block definition can be stored as a single file (.cxf) for reuse in other projects.	

Number of Steps Used by Function Blocks

When function blocks are used, memory is required for the following two items:

1. Function block definitions
2. Instance generation processing when function blocks are pasted into the user program as function block instances

Therefore, the number of steps used in memory will increase with the number of instances of function blocks created in the program (item 2).

Memory Areas Used for Function Blocks

The area of memory used for function blocks depends on the model of CJ2 CPU Unit that is used, as described in the following table.

The CJ2M CPU Units have a special area called the FB Program Area to store function block definitions. The CJ2H CPU Units do not have this area.

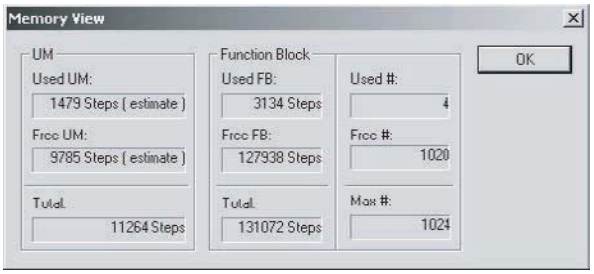
CPU Unit	Models	Memory areas used for function blocks
CJ2H	CJ2H-CPU6□-EIP CJ2H-CPU6□	User Program Area
CJ2M	CJ2M-CPU3□ CJ2M-CPU1□	Creating instances of function blocks (i.e., calling function block definitions and transferring parameters) uses user program memory. Function block definitions use memory in the FB Program Area, a special area for function blocks. If the capacity of the FB Program Area is exceeded, the user program area is used.

Checking Function Block Memory Usage

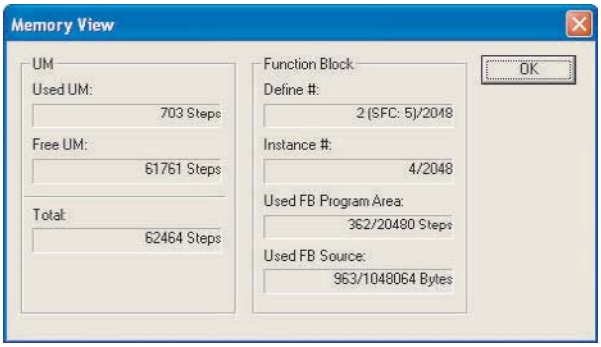
Select ***View*** – ***Memory View*** from CX-Programmer version 9.0 or higher.

The following Memory View Dialog Box will be displayed.

● When the PLC Model ("Device Type") Is a CS/CJ-series PLC (Unit Version 3.0 or Later) Other Than the CJ2M



● When the PLC Model ("Device Type") Is the CJ2M



Additional Information

Refer to the *CX-Programmer Operation Manual* (Cat. No. W446) for details on the Memory View Dialog Box.

Calculating the Number of Steps Used by Function Blocks

Use the following formula as a guide to the number of steps used by function blocks.

Number of steps used

= Number of instances × (Call part size m + I/O parameter transfer part size n × Number of parameters) + Number of instruction steps in the function block definition p (See note.)

Note Memory will not be used for the number of instruction steps in the function block definition (p) in the second and later instances when the same function block definition is created in multiple locations (i.e., for multiple instances). Therefore, in the above equation, the number of instruction steps in the function block definition (p) is not multiplied by the number of instances.

Items			Number of steps
m	Call part		57 steps
n	I/O parameter transfer part The data types are indicated in parentheses.	1-bit (BOOL) input symbol or output symbol	6 steps
		1-word (INT, UINT, WORD) input symbol or output symbol	6 steps
		2-word (DINT, UDINT, DWORD, REAL) input symbol or output symbol	6 steps
		4-word (LINT, ULINT, LWORD, LREAL) input symbol or output symbol	12 steps
		I/O symbols	18 steps
p	Number of instruction steps in function block definition	The total number of instruction steps (same as standard user program) + 27 steps.	

Example

Five input symbols with a 1-word (INT) data type, five output symbols with a 1-word (INT) data type, and a function block definition with 100 steps:

Number of steps for 1 instance = 57 + (5 + 5) × 6 steps + 100 steps + 27 steps = 244 steps

If the function block is written in the standard text language, the actual number of steps cannot be calculated. The number of instruction steps in a function block definition can be checked in the properties of the function block definition.



Additional Information

For function block specifications, notation methods, and input procedures, refer to the C*X-Programmer Operation Manual: Function Blocks and Structured Text* (Cat. No. W447).

5-5 Symbols

5-5-1 Overview

Symbols

● Symbol Tables

I/O memory area addresses or constants can be specified using character strings by registering the character strings as symbols. Register the symbols in the symbol tables of the CX-Programmer. Programming with symbols enables programming with names rather than having to be aware of the actual addresses.

Symbol tables are saved in a CX-Programmer project file (.CXP) along with other parameters, such as the user program and I/O tables.

● Conditions for Using Symbols

Whether using symbols is required or optional depends the programming language as well as whether the symbol is used inside or outside of a function block, as given in the following table.

Program element	Programming language	Specifying symbols
Programming outside function blocks	Ladder diagram	Optional
	ST	Required
	SFC	Optional
Programming inside function blocks (Inside function blocks, symbols are called “variables.”)	Ladder diagram	Required
	ST	

5-5-2 Types of Symbols

The following types of symbols can be used.

Program Symbols

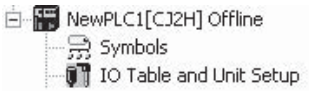
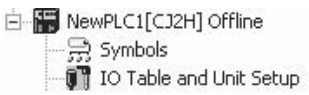
● Types of Symbols

- Global Symbols
  Global symbols can be accessed from any task in the CPU Unit.
- Local Symbols
  Local symbols can be accessed from only one task.
- Network Symbols (CJ2H-CPU6□-EIP or CJ2M-CPU3□ only)
  Tags can be used from exterior devices to access the I/O memory of the local CPU Unit through the network symbols defined in the CPU Unit. Network symbols can also be used in the local CPU Unit.

● Address Allocation

Addresses are allocated to symbols used in programming based on the following factors.

- User Specifications
- Automatic Allocation using the CX-Programmer
  The area of memory used for automatic allocations is set by selecting ***Memory Allocation*** – ***Automatic Address Allocation*** from the PLC Menu in the CX-Programmer.

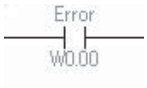
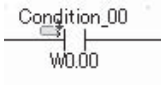
Type of symbol	Name	CX-Programmer project tree	Scope			Address and I/O comment (without a symbol name)
			Access using network symbols	Access from other tasks	Access from the local task	
Program symbols	Global symbols	PLC tree 	Not supported	Supported	Supported	Supported
	Local symbols	Program tree 		Not supported	Supported	Not supported
	Network symbols	Global variable table in PLC tree 	Supported	Supported	Supported	Not supported

Note “Global” and “local” indicate only the scope of application of the symbol. It has nothing to do with the scope of application for the memory address. Therefore, a warning but not an error will occur in the following cases, and it will be possible to transfer the user program.

- The same addresses is used for two different local symbols
- The same addresses is used for a global symbol and a local symbol

Additional Information

In CX-Programmer programs, global symbols, local symbols, and network symbols can be identified by the following character colors and icons indicating symbols.

Type of symbol	Display color	Example (default color)
Global symbols	Black (fixed)	
Local symbols	Blue (default) Select ***Tools*** – ***Options***, display the **Appearance** Tab Page, and select the local symbols item to change the color.	
Network symbols	Black (fixed) An icon ( ) indicating a network symbol will be displayed under the symbol name.	

Variables in Function Blocks

Variables can be accesses only from the algorithms in function blocks.

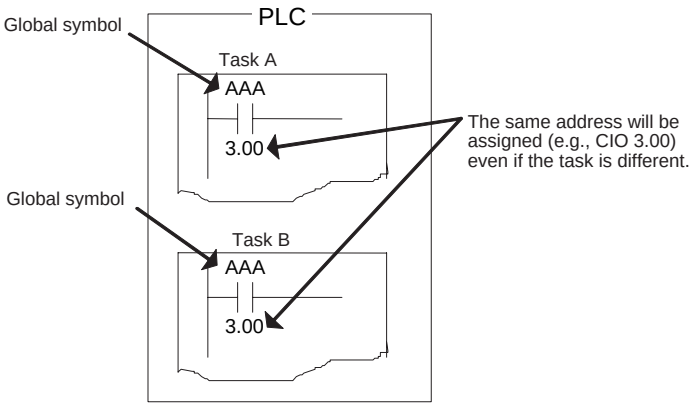
Name	CX-programmer project tree	Scope		Address and I/O comment (without a symbol name)
		Access using network symbols	Access from outside the function block	
Variables (i.e., symbols used in function blocks)	Definitions in function blocks 	Not supported (Network symbols cannot be registered for external variables.)	Not supported	Not supported

Some of the specifications for variables are different from those for symbols used outside of function blocks. Set the area to be used for variables by selecting *Memory Allocation – Function Block/SFC Memory* from the PLC Menu in the CX-Programmer.

5-5-3 Global Symbols

Global symbols are symbols that are supported for all tasks in the target CPU Unit. For example, therefore, a symbol named “AAA” would be the same address in all tasks in the target CPU Unit.

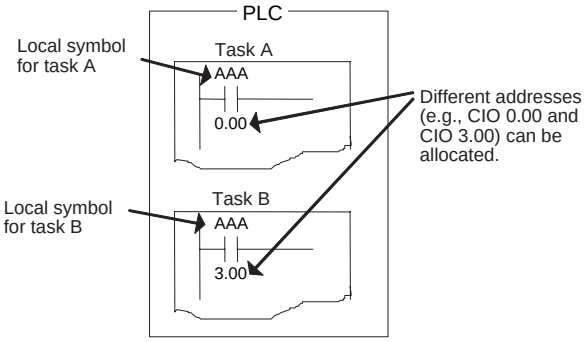
Example: If the symbol named “AAA” is set as a global symbol, the same address will be assigned (e.g., CIO 3.00) even if the task is different.



5-5-4 Local Symbols

Local symbols can be accessed only from the task they are in defined. Each local symbol is supported in only one task. Local symbols are specified separately for each task, so identical symbols will be handled as different symbols for different tasks.

For example, if the symbol named “AAA” is set as a local symbol, different addresses (e.g., CIO 0.00 and CIO 3.00) can be allocated for different tasks with the same “AAA” symbol.



5-5-5 Network Symbols (CJ2H-CPU6□-EIP and CJ2M-CPU3□ Only)

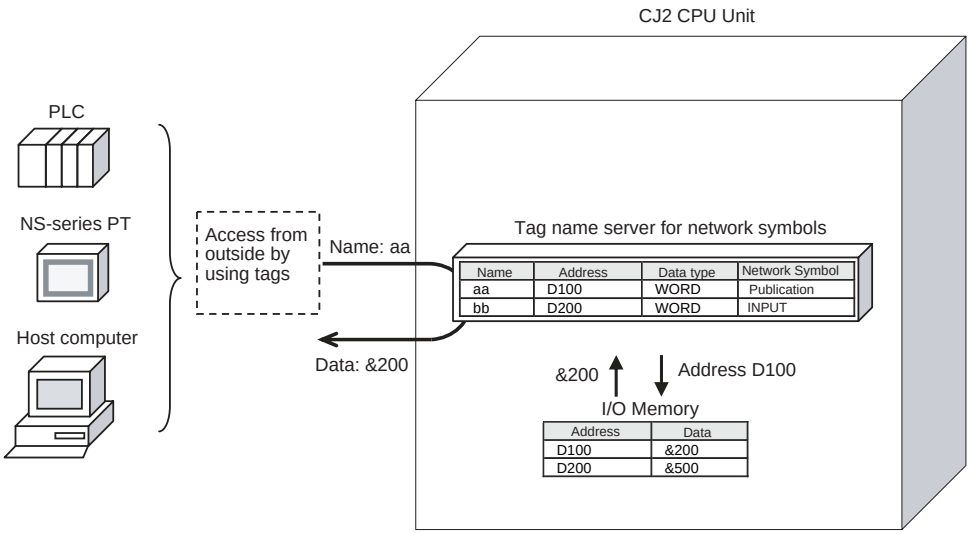
Overview

Tags can be used from exterior devices to access the I/O memory of the local CPU Unit through the network symbols defined in the CPU Unit. The CJ2 CPU Units have an internal tag name server that can convert network symbols to actual I/O addresses, and so it is possible to access the CPU Unit's I/O memory from the outside by using tags to access network symbols.

Network symbols can be used for the following applications.

- Data links with other PLCs on EtherNet/IP
- Tag access from outside with host or PT.

Set the network symbols in the global symbols tables of the CX-Programmer.



Additional Information

Refer to the *EtherNet/IP Units Operation Manual* (Cat. No. W465) for the procedures to create tag data links.

Setting Network Symbols

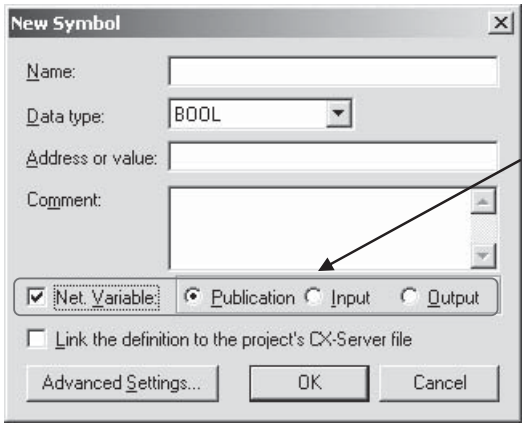
Use the following procedure to set network symbols.

1 Right-click on the global variable table and select ***Insert Symbol***.

Note Network symbols cannot be specified in the local symbol tables.

2 The following New Symbol Dialog Box will be displayed.

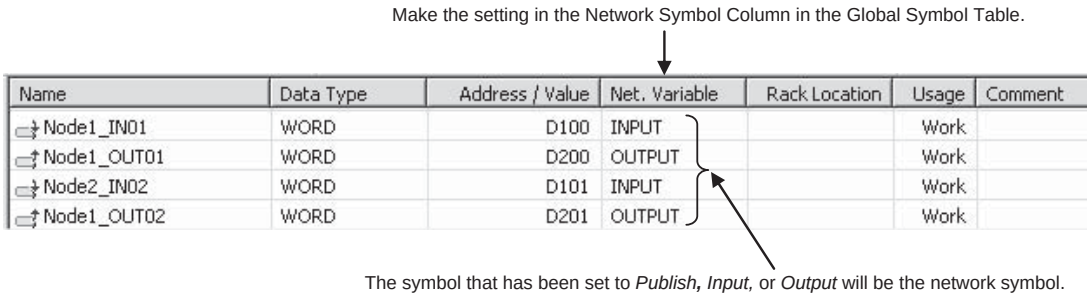
Select the Net, Variable Check Box, and then select the *Publication, Input, or Output* Option, and then click the **OK** Button.



A maximum of 48 characters can be used for a network symbol name. The names are not case sensitive.

Type of symbol	Description	Network symbol column in global symbol table	Description
Network symbol	Symbols accessible from outside by using tags.	Publication	• Symbols accessible form outside by using CIP message communications. • Tag data links are not supported.
		Input	• Symbols accessible from outside by using CIP message communications or tag data links. • Symbols for data input (from other CPU Unit to local CPU Unit) when tag data links are used.
		Output	• Symbols accessible from outside by using CIP message communications or tag data links. • Symbols for data output (from local CPU Unit to other CPU Unit) when tag data links are used.

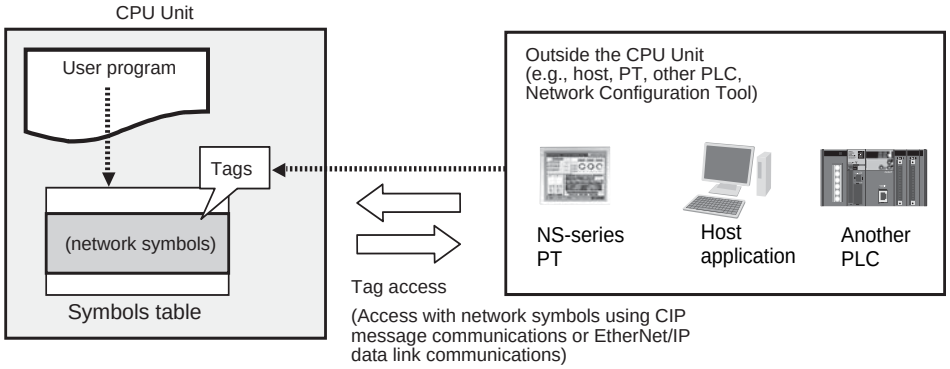
● Network Symbol Column in Global Symbol Table



Network Symbol Usage Conditions and Handling

● Handling Network Symbols from the Outside

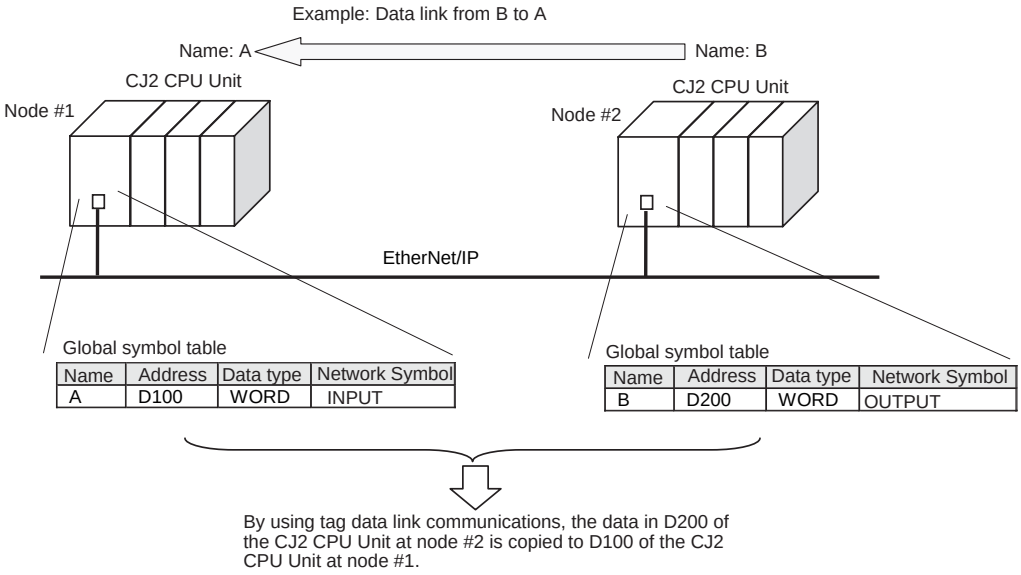
Symbols set as network symbols in the CPU Unit are recognized as tags from outside the PLC (such as by a host, PT, Network Configuration Tool, or other PLCs).



Network Symbol Applications

● Using Network Symbols as EtherNet/IP Data Link Tags

The data in a remote PLC can be specified with tags by using EtherNet/IP tag data link communications. Set to the network symbol to *Input* or *Output* in the Network Symbol Column in the Global Symbol Table.

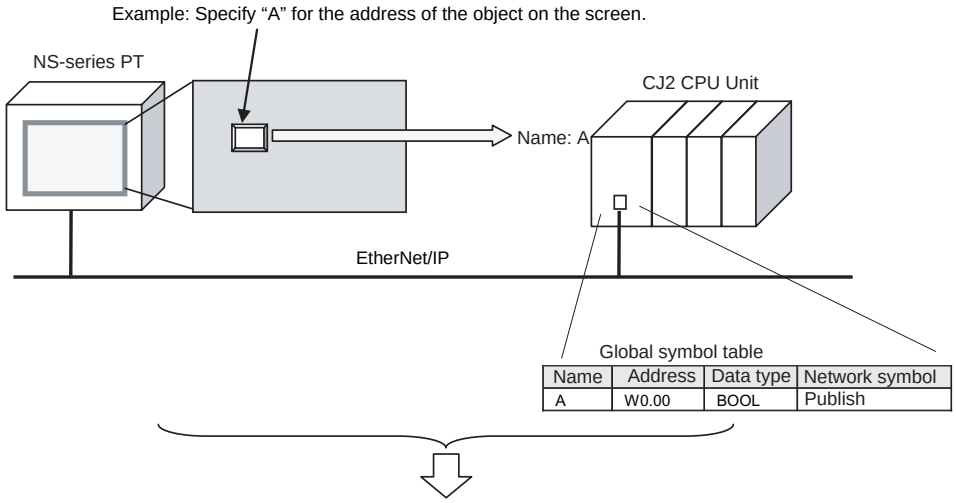


Name	Address	Data type	Network Symbol
A	D100	WORD	INPUT

Name	Address	Data type	Network Symbol
B	D200	WORD	OUTPUT

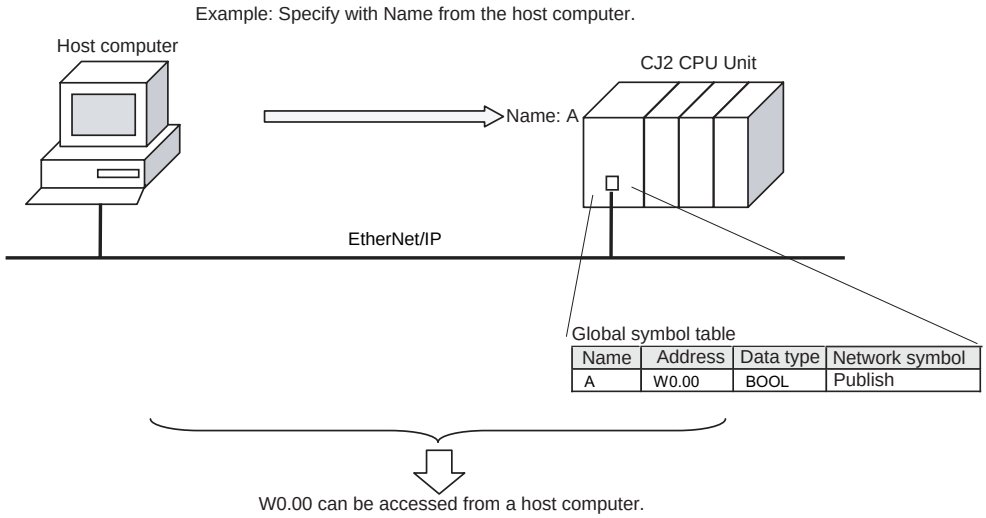
Using Tags in Communications with an NS-series PT

The data in a remote PLC can be specified with tags by using objects on the display of the NS-series PT. Set the network symbol to *Publication* in the *Network Symbol* Column in the Global Symbol Table.



Using Tags in CIP Message Communications from the Host Computer

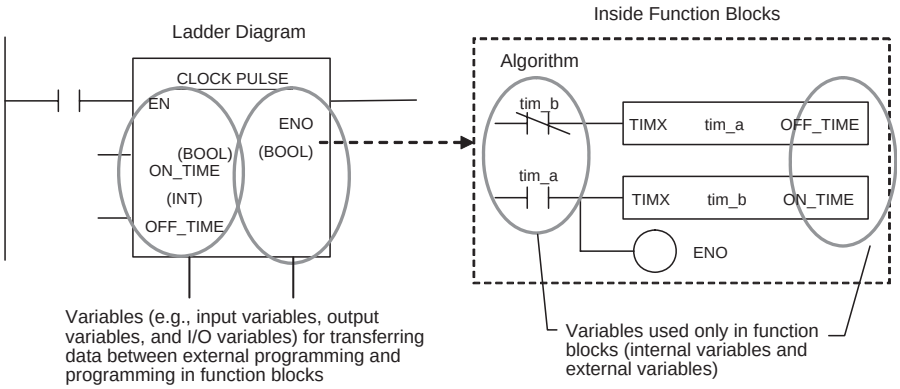
Data in a remote PLC can be specified with tags from a host computer. Set the network symbol to *Publish* in the global symbol table.



5-5-6 Variables in Function Blocks

Programs in function blocks are all written with variables rather than actual addresses. Variables in function blocks have different variable types and specifications than symbols outside of function blocks. For details on variables in function blocks, refer to the C*X-Programmer Operation Manual: Function Blocks and Structured Text* (Cat. No. W447).

● Function Block Variable Fields



5-5-7 Symbol Data Types

It is possible to specify the data type for addresses allocated for symbols.

Data Types That Can Be Set for Symbols

The data types that can be specified inside function blocks may be different from the types that be specified outside function blocks. The data types that can be used for each are specified in the following table.

Data type	Symbols in programming	Variable in function blocks
BOOL (bit)	OK	OK
UINT (one-word unsigned binary)	OK	OK
UDINT (two-word unsigned binary)	OK	OK
ULINT (four-word unsigned binary)	OK	OK
INT (one-word signed binary)	OK	OK
DINT (two-word signed binary)	OK	OK
LINT (four-word signed binary)	OK	OK
UINT BCD (one-word unsigned binary)*1	OK	No
UDINT BCD (two-word signed binary)*1	OK	No
ULINT BCD (four-word signed binary)*1	OK	No
REAL (two-word floating point)	OK	OK
LREAL (four-word floating point)	OK	OK
CHANNEL (word)*1	OK	No
NUMBER (constant or number)*2	OK	No
WORD (one-word hexadecimal)	OK	OK
WORD (two-word hexadecimal)	OK	OK
LWORD (four-word hexadecimal)	OK	OK
STRING (character string: 1 to 255 ASCII characters)	OK	OK
TIMER	No	OK
COUNTER	No	OK
User-defined data types (data structures)	OK	OK*3

*1 Data types will be converted as follows in ST programming.

- UINT BCD: WORD
- UDINT BCD: DWORD
- ULINT BCD: LWORD
- CHANNEL: WORD

*2 These data types cannot be used in ST programming. A program error will occur.

*3 Only internal symbols in function blocks written with ladder diagrams.

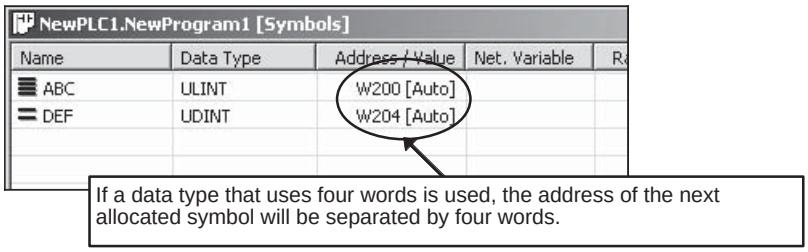
Applications of Data Types

Data types set for symbols can be used for the following functions.

● Application Using Automatic Address Allocation

Addresses can be allocated according to the data type set for the symbol when automatic address allocation is used with the CX-Programmer.

Example: Setting the ULINT (4-word Unsigned Binary) Data Type



Specifying Arrays

Arrays can be specified as symbol attributes (one-dimensional arrays only). Arrays can be specified for all data types except for STRING and NUMBER.

Use an array specification if multiple symbols with the same data attribute are managed as one group. Use the following procedure if array specification is used.

1 Enter the symbol name in the New Symbol Dialog Box or Edit Symbol Dialog Box, and then click the **Advanced Settings** Button. The Advanced Settings Dialog Box will be displayed.

2 Select the *Array Symbol* Option in the *Array Settings* Field, and then enter the maximum number of elements.

When writing an array variable, specify a suffix between brackets after the variable name.

Example: If the symbol name is PV and the maximum number of elements is 3, specify array variables as PV[0], PV[1], and PV[2] in instruction operands.

The suffix of the array is specified with an element number starting from 0. The element number can be entered directly, or it can also be specified indirectly by inputting a symbol or memory address.

⚠ Caution

If a symbol or memory address (only symbols are allowed for ST programming) is specified for the suffix of an array variable in ladder or ST programming to indirectly specify the element number, be sure that the element number does not exceed the maximum memory area range.

Specifying a element number that exceeds the maximum range of the memory area specified for the symbol will result accessing data in a different memory area, and may result in unexpected operation.

Array variables with suffixes that are word addresses or symbols cannot be used as operands in the immediate refresh version of an instruction.

User-defined Data Types (Data Structures)

When using CX-Programmer version 9.0 or higher with a CJ2 CPU Unit, you can create data structures as user-defined data types.

● Data Structures

A data structure is a user-defined data type that groups more than one data type. Names can be assigned to the data types. The name of the variable that uses a user-defined data type is specified along with the name of one of the variables within the data structure. The overall data structure is called a structure variable and the variables within the data structure are called members.

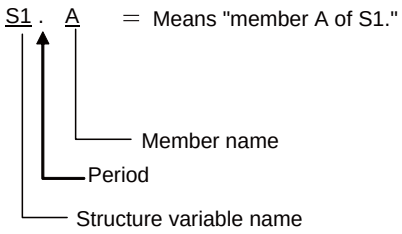
Additional Information

- Arrays can also be used to handle multiple pieces of data. An array data type, however, is different from a data structure in that it contains data with the same data type that is accessed by specifying an offset from the beginning of the array. With a data structure, data with the same or with different data types is accessed using member names. Also, with the CX-Programmer, structure data types can be assigned names.
- Structure variables can be placed in arrays.
- Also, array variables can be used as members of data structures.

● Ladder Program Notation and Input for Structure Variables

In a Ladder Program, the structure variable name and member name are separated by a period.

Example:

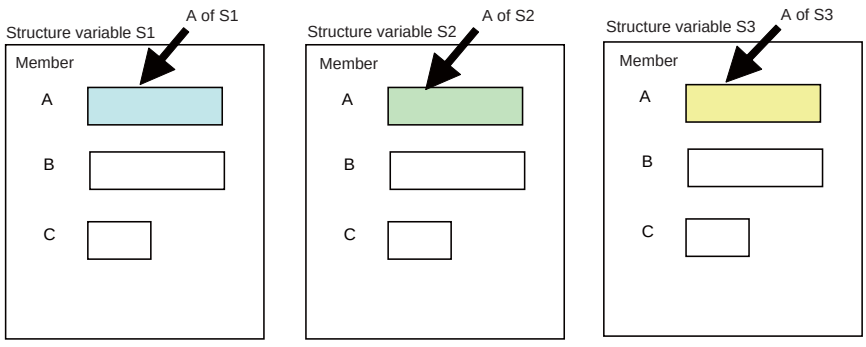


● Applications of Data Structures

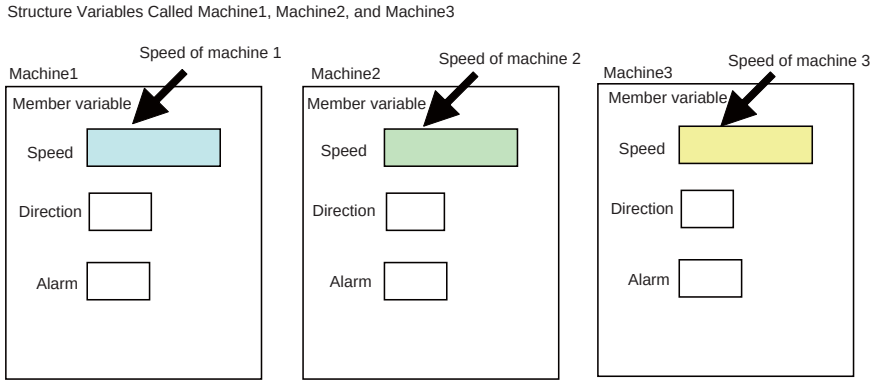
When you specify data that is grouped, you specify a member of a specific group. In other words, you can specify data in a hierarchy in the form "main - sub."

Using Data Structures without Arrays

Example for Structure Variables S1, S2, and S3 Each with Members A, B, and C



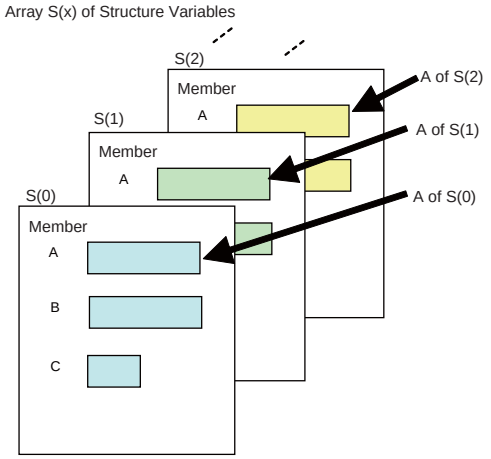
For example, the same member (e.g., the speed) can be specified for each machine.



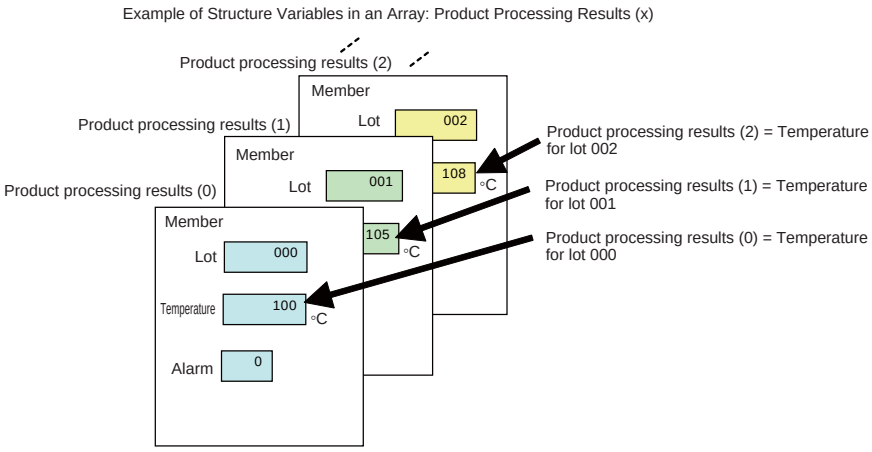
Combining Data Structures with Arrays

Placing Structure Variables in an Array

When there is a large volume of data in the same form, as with recipe data for different products, structure variables can be placed in an array. This is used to create a database. In this case, the structure variable becomes one record and each member becomes a field in the database.

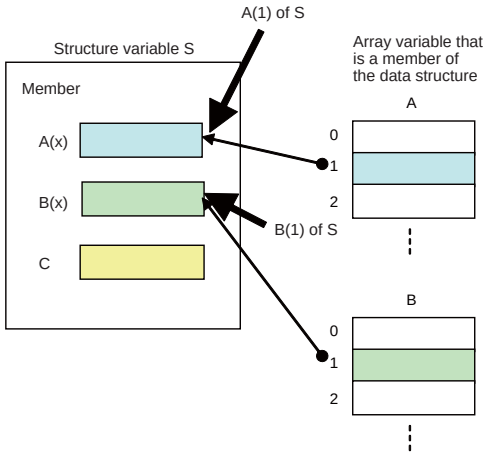


For example, this can be used to specify the processing results for each production lot.

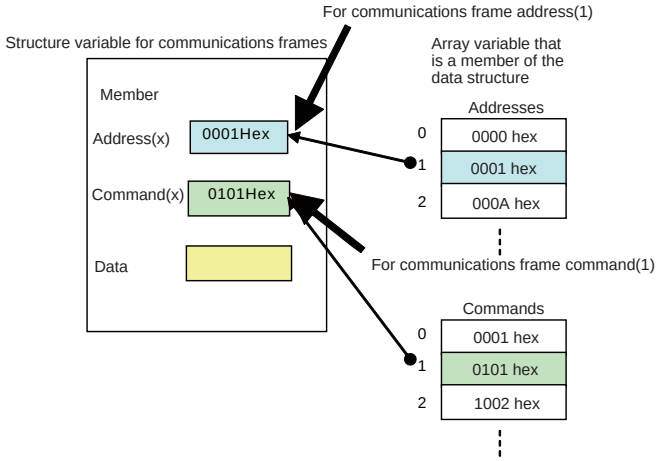


Using Array Variables as Members of Data Structures

Arrays can be used as members of a data structure when there is a specific overall structure to the data with members that each contains multiple elements in library fashion. Members can be freely specified from the arrays. This is used when data is built from libraries.



Example: Communications frames can be created by selecting elements from libraries.



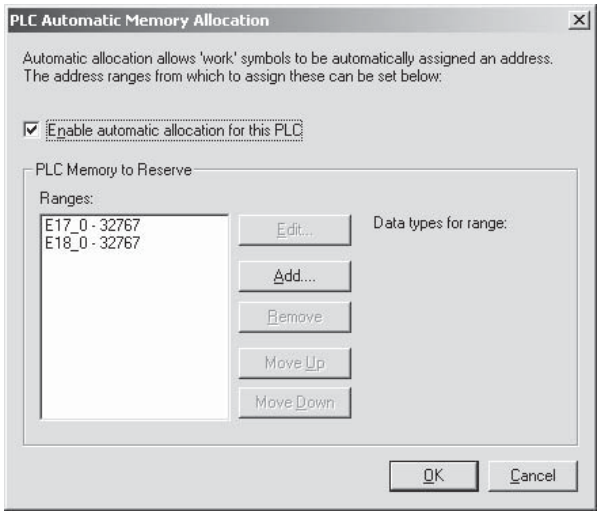
For detailed information on data structure specifications and creation procedures, refer to the *CX-Programmer Operation Manual: Function Blocks and Structured Text* (Cat. No. W447).

5-5-8 Automatic Address Allocation to Symbols

When programming, you can automatically allocate addresses to symbols. Select ***Memory Allocation*** – ***Automatic Address Allocation*** from the PLC Menu in the CX-Programmer and then set the range and data type for automatic allocation. With CJ2 CPU Units, addresses in the CIO Area, Auxiliary Area, Holding Area, Work Area, DM Area, or EM Area can be allocated.

When automatically allocating symbol addresses in the EM Area, we recommend used the required number of banks starting backward from the highest EM Area bank.

The following example shows the settings to automatically allocate symbol addresses in banks 17 and 18 hex of a CJ2H-CPU68-EIP CPU Unit (which has EM Area banks from 00 to 18 hex).



Refer to the *CX-Programmer Operation Manual* (Cat. No. W446) for details on the area settings for automatic address allocation.

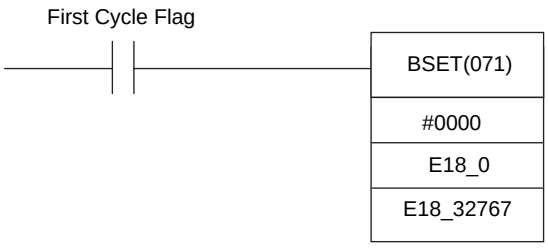
Additional Information

- With CJ2H CPU Units, bits in EM Area banks that are set for automatic address allocation can be force-set/reset. Refer to *6-14 Extended Data Memory Area* for the banks in which bits can be force-set/reset.
- Automatic address allocation is not possible for structure variables.

Precautions for Safe Use

Data in the EM Area is backed up when the power supply is turned OFF or the operating mode is changed. Be careful when using output bits specified as BOOL data. If necessary, including programming to clear memory as required.

Example: The following instructions can be used to clear bank 18 hex to all zeros when power is turned ON when EM Area bank 18 hex is set for automatic address allocation.

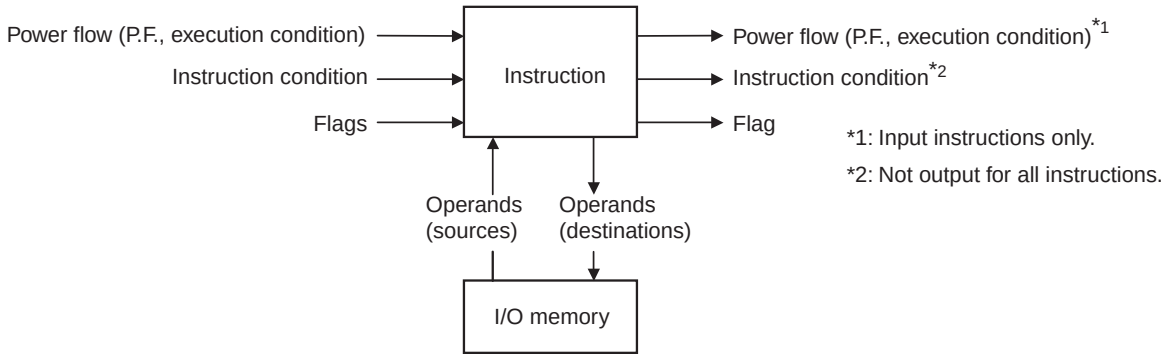


5-6 Instructions

5-6-1 Basic Understanding of Instructions

Structure of Instructions

Programs consist of instructions. The conceptual structure of the inputs to and outputs from an instruction is shown in the following diagram.

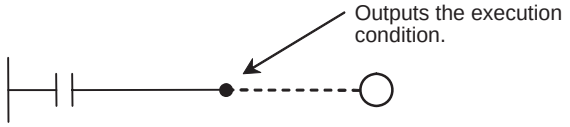


● Power Flow

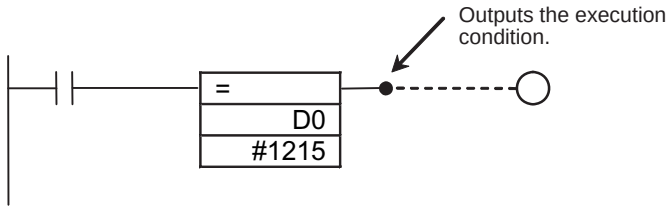
The power flow is the execution condition that is used to control the execute and instructions when programs are executing normally. In a ladder program, power flow represents the status of the execution condition.

Input Instructions

- Load instructions indicate a logical start and outputs the execution condition.

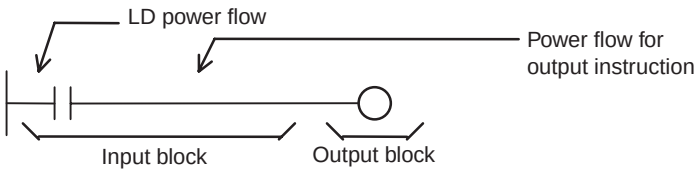


- Intermediate instructions input the power flow as an execution condition and output the power flow to an intermediate or output instruction.



Output Instructions

Output instructions execute all functions, using the power flow as an execution condition.



Instruction Conditions

Instruction conditions are special conditions related to overall instruction execution that are output by the following instructions. Instruction conditions have a higher priority than power flow (P.F.) when it comes to deciding whether or not to execute an instruction. An instruction may not be executed or may act differently depending on instruction conditions. Instruction conditions are reset (canceled) at the start of each task, i.e., they are reset when the task changes.

The following instructions are used in pairs to set and cancel certain instruction conditions. These paired instructions must be in the same task.

Instruction condition	Description	Setting instruction	Canceling instruction
Interlocked	An interlock turns OFF part of the program. Special conditions, such as turning OFF output bits, resetting timers, and holding counters are in effect.	IL(002)	ILC(003)
BREAK(514) execution	Ends a FOR(512) - NEXT(513) loop during execution. (Prevents execution of all instructions until to the NEXT(513) instruction.)	BREAK(514)	NEXT(513)
	Executes a JMP0(515) to JME0(516) jump.	JMP0(515)	JME0(516)
Block program execution	Executes a program block from BPRG(096) to BEND(801).	BPRG(096)	BEND(801)

Flags

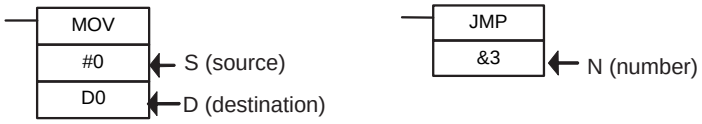
In this context, a flag is a bit that serves as an interface between instructions.

Input flags		Output flags	
Flag	Description	Flag	Description
Carry (CY) Flag	The Carry Flag is used as an unspecified operand in data shift instructions and addition/subtraction instructions.	Condition Flags	Condition Flags include the Always ON/OFF Flags, as well as flags that are updated by results of instruction execution. In user programs, these flags can be specified by labels, such as P_On, P_Off, P_ER, P_CY, P_EQ rather than by addresses.
Flags for Special Instructions	These include teaching flags for FPD(269) instructions and network communications enabled flags.	Flags for Special Instructions	These include memory card instruction flags and MSG(046) execution completed flags.

● Operands

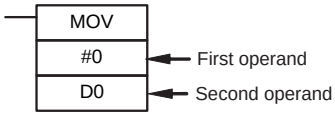
Operands specify preset instruction parameters (boxes in ladder diagrams) that are used to specify I/O memory area contents or constants. An instruction can be executed entering an address or constant as the operands. Operands are classified as source, destination, or number operands.

Example



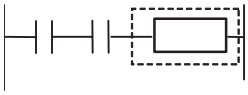
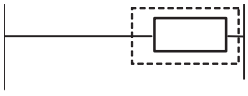
Operand types		Operand symbol	Description	
Source	Specifies the address of the data to be read or a constant.	S	Source Operand	Source operand other than control data (C)
		C	Control data	Compound data in a source operand that has different meanings depending bit status.
Destination (Results)	Specifies the address where data will be written.	D	---	
Number	Specifies a particular number used in the instruction, such as a jump number or subroutine number.	N	---	

Note Operands are also called the first operand, second operand, and so on, starting from the top of the instruction.



Instruction Location and Execution Conditions

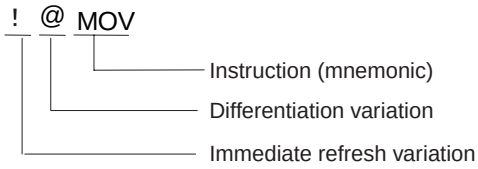
The following table shows the possible locations for instructions. Instructions are grouped into those that do and those do not require execution conditions. Refer to *A-1 Instruction Functions* for details on instructions.

Instruction		Location	Execution condition	Diagram	Examples
Input instructions	Logical start (Load instructions)	Connected directly to the left bus bar or is at the beginning of an instruction block.	Not required.		LD, LD TST(350), LD > (and other symbol comparison instructions)
	Intermediate instructions	Between a logical start and the output instruction.	Required		AND, OR, AND TEST(350), AND > (and other ADD symbol comparison instructions), UP(521), DOWN(522), NOT(520), etc.
Output instructions		Connected directly to the right bus bar.	Required		Most instructions including OUT and MOV(021).
			Not required.		END(001), JME(005), FOR(512), ILC(003), etc.

Instruction Variations

The following variations are available for instructions to differentiate executing conditions and to refresh data when the instruction is executed (immediate refresh).

Variation		Symbol	Description
Differentiation	ON	@	Instruction that differentiates when the execution condition turns ON.
	OFF	%	Instruction that differentiates when the execution condition turns OFF.
Immediate refreshing		!	Refreshes data in the I/O area specified by the operands or the Special I/O Unit words when the instruction is executed.



Execution Conditions

The following two types of basic and special instructions can be used.

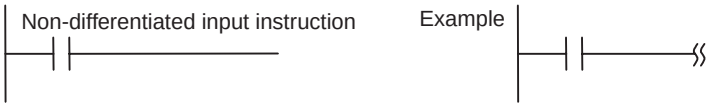
- Non-differentiated instructions: Executed every cycle
- Differentiated instructions: Executed only once

● Non-differentiated Instructions

Output instructions (Instructions That Require Input Conditions):
These instructions are executed once every cycle while the execution conditions are satisfied (ON or OFF).



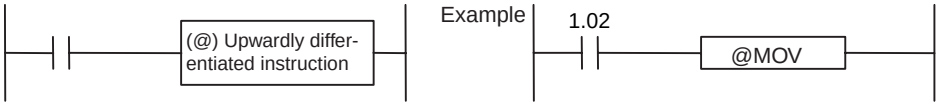
Input Instructions (Logical Starts and Intermediate Instructions):
These instructions read bit status, make comparisons, test bits, or perform other types of processing every cycle. If the results are ON, power flow is output (i.e., the execution condition is turned ON).



● Input-differentiated Instructions

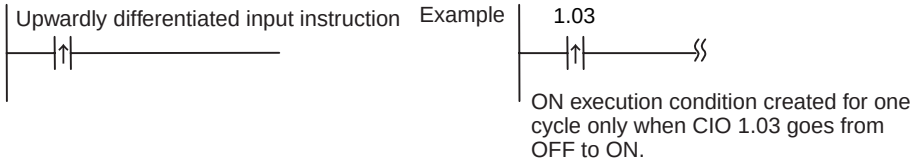
Upwardly Differentiated Instructions (Instruction Preceded by @)

Output Instructions:
The instruction is executed only during the cycle in which the execution condition changed from OFF to ON and are not executed in the following cycles.

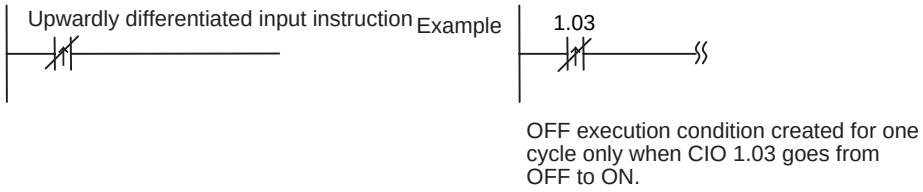


Executes the MOV instruction once when CIO 1.02 goes OFF → ON.

Input Instructions (Logical Starts and Intermediate Instructions):
The instruction reads bit status, makes comparisons, tests bits, or perform other types of processing every cycle and will output an ON execution condition (power flow) when results switch from OFF to ON. The execution condition will turn OFF the next cycle.

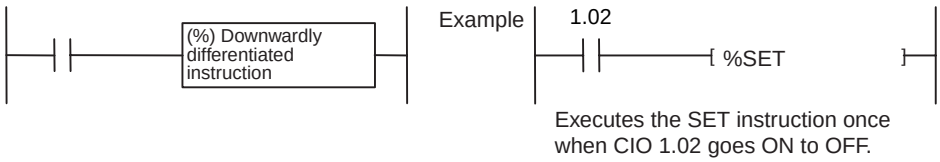


Input Instructions (Logical Starts and Intermediate Instructions):
The instruction reads bit status, makes comparisons, tests bits, or perform other types of processing every cycle and will output an OFF execution condition (power flow stops) when results switch from OFF to ON. The execution condition will turn ON the next cycle.

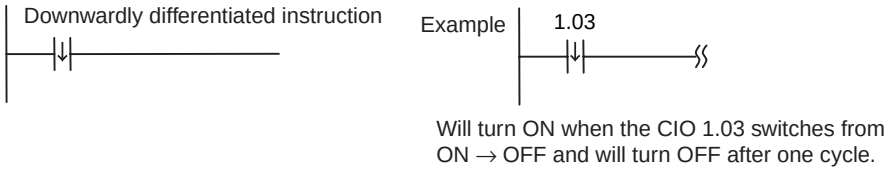


Downwardly Differentiated Instructions (Instruction Preceded by %)

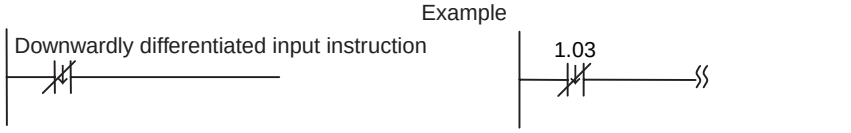
Output Instructions:
The instruction is executed only during the cycle in which the execution condition changed from ON to OFF and is not executed in the following cycles.



Input Instructions (Logical Starts and Intermediate Instructions):
The instruction reads bit status, makes comparisons, tests bits, or perform other types of processing every cycle and will output the execution condition (power flow) when results switch from ON to OFF. The execution condition will turn OFF the next cycle.

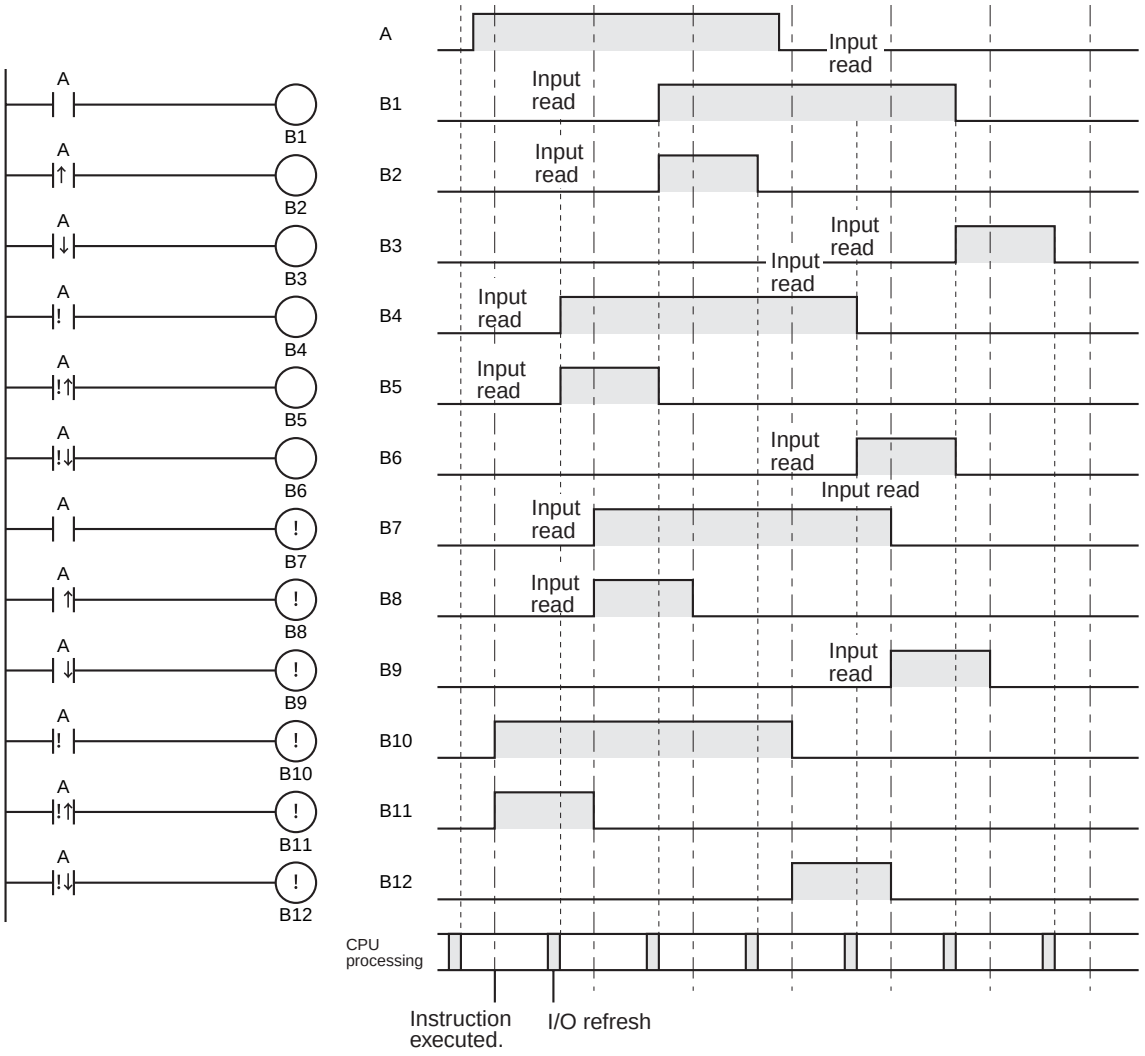


Input Instructions (Logical Starts and Intermediate Instructions):
The instruction reads bit status, makes comparisons, tests bits, or perform other types of processing every cycle and will output an OFF execution condition (power flow stops) when results switch from ON to OFF. The execution condition will turn ON the next cycle.



I/O Instruction Timing

The following timing chart shows different operating timing for individual instructions using a program comprised of only LD and OUT instructions.

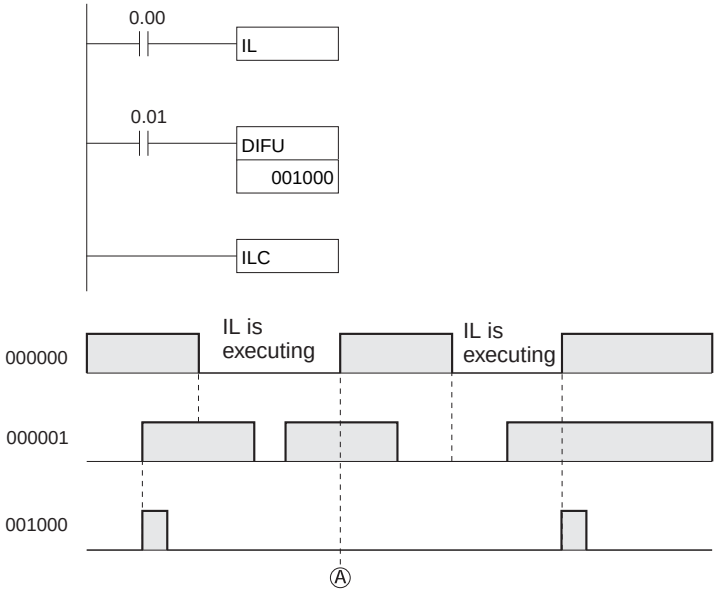


● Differentiated Instructions

- A differentiated instruction has an internal flag that tells whether the previous value is ON or OFF. At the start of operation, the previous value flags for upwardly differentiated instruction (DIFU and @ instructions) are set to ON and the previous value flags for downwardly differentiated instructions (DIFD and % instructions) are set to OFF. This prevents differentiation outputs from being output unexpectedly at the start of operation.
- An upwardly differentiated instruction (DIFU or @ instruction) will output ON only when the execution condition is ON and flag for the previous value is OFF.

Using Differentiated Instructions in Interlocks (IL - ILC Instructions)

In the following example, the previous value flag for the differentiated instruction maintains the previous interlocked value and will not output a differentiated output at point A because the value will not be updated while the interlock is in effect.



Using Differentiated Instructions in Jumps (JMP(004) - JME(005) Instructions)

Just as for interlocks, the previous value flag for a differentiated instruction is not changed when the instruction is jumped, i.e., the previous value is maintained.

- With downwardly differentiated instructions (DIFD(014) or instructions with a %), outputs will turn ON when inputs turn OFF only when the previous value flag is ON.
- With both upwardly and downwardly differentiated instructions, outputs will turn OFF in the next cycle.

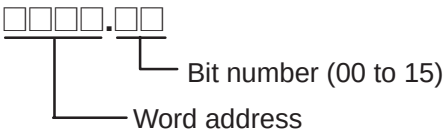
Precautions for Correct Use

Do not use the Always P_On Flag or A200.11 (First Cycle Flag) as the input bit for an upwardly differentiated instruction. Do not use the Always P_Off Flag as the input bit for a downwardly differentiated instruction. If either is used, the instruction will never be executed.

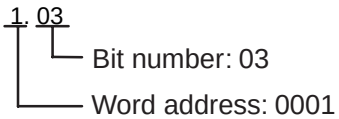
5-6-2 Specifying Operands

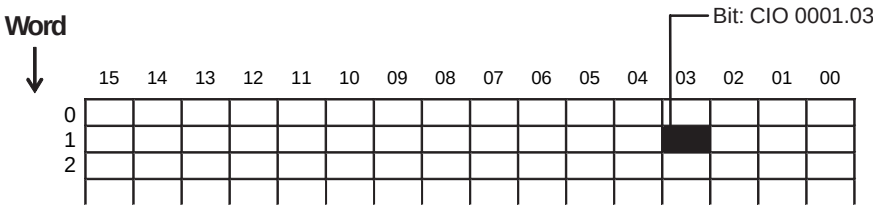
Addressing I/O Memory Areas

- **Bit Addresses**



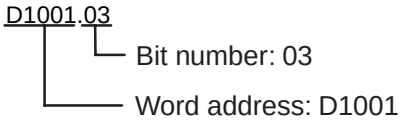
Example: The address of bit 03 in word 1 in the CIO Area would be as shown below.

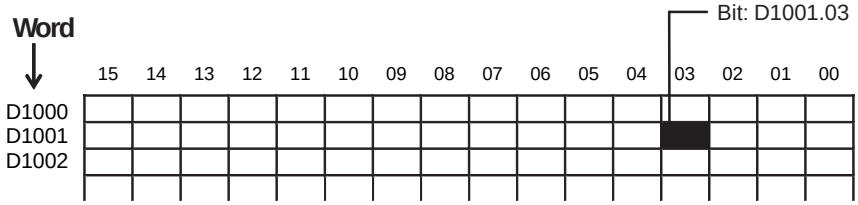




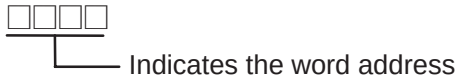
With the CJ2 CPU Unit, bit addresses can be specified in the DM and EM Areas.

Example: DM area

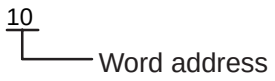




Word Addresses

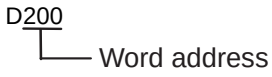


Example: I/O Area

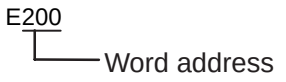


DM and EM Areas addresses are given “D” or “E” prefixes, as shown below for the address D200.

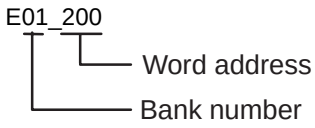
Example: DM Area



Example: EM Area



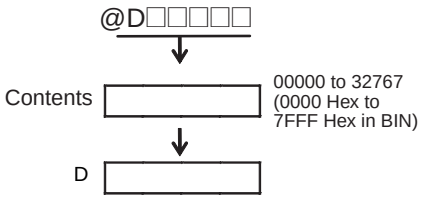
Example: EM Area Bank 1



Specifying Operands

Operand	Description	Notation	Application examples
Specifying bit addresses	The word and bit numbers are specified directly to specify a bit (input bits). □□□□.□□ Bit number (00 to 15) Indicates the word address. *1	1.02 Bit number (02) Word number: 1	1.02
Specifying word addresses	The word number is specified directly to specify the 16-bit word. □□□□ Indicates the word address.	3 Word number: 3 D200 Word number: 200	MOV 3 D200
Specifying offsets for bit addresses	In brackets, specify the number of bits to offset the specified starting bit address. □□□□.□□ [□□] Offset Constant 0 to 15 or word address in I/O memory Starting bit address A symbol can also be specified for the starting bit address. Only Holding, Work, DM, and EM Area addresses can be used regardless of whether a physical address or symbol is used. A constant or word address in I/O memory can be used for the offset. If a word address is specified, the contents of the word is used directly as the offset.	10.00[2] Number of bits to offset the address → 10.02 Starting bit address 10.00[WO] Number of bits to offset the address (W0 = &2) → 10.02 Starting bit address	10.00[2]
Specifying offsets for word addresses	In brackets, specify the number of bits to offset the specified starting bit address. □□□□.□□ [□□] Offset Constant 0 to 15 or word address in I/O memory Starting word address A symbol can also be specified for the starting word address. Only Holding, Work, DM, and EM Area addresses can be used regardless of whether a physical address or symbol is used. A constant or word address in I/O memory can be used for the offset. If a word address is specified, the contents of the word is used directly as the offset.	D0[2] Number of words to offset address → D2 Starting word address D0[WO] Number of words to offset address (W0 = &2) → D2 Starting word address	MOV 3 D0[200]

*1 The same addresses are used to access timer/counter Completion Flags and Present Values. There is also only one address for a Task Flag.

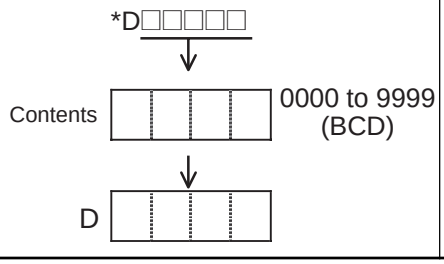
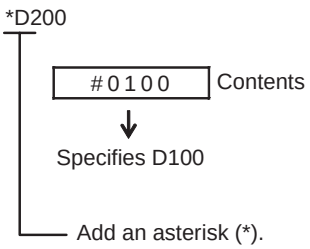
Operand	Description	Notation	Application examples
Specifying indirect DM/EM addresses in Binary Mode*2	The offset from the beginning of the area is specified. The contents of the address will be treated as binary data (00000 to 32767) to specify the word address in DM Area or EM Area. Add the @ symbol at the front to specify an indirect address in Binary Mode. 		
	D0 to D32767 are specified if @D□□□□□ contains 0000 to 7FFF hex (0 to 32767 decimal).	@D300 &256 Contents Hex: #0100 Specifies D256. Add the @ symbol.	MOV #0001 @D300
	E0 _0 to E0 _32767 of bank 0 in the EM Area are specified if @D□□□□□ contains 8000 to FFFF hex (32768 to 65535 decimal).	@D300 &32769 Contents Hex: #8001 Specifies E0_01.	
	E□_0 to E□_32767 in the specified bank are specified if @E□_□□□□□ contains 0000 to 7FFF hex (0 to 32767 decimal).	@E1_200 &257 Contents Hex: #0101 Specifies E1_257.	MOV #0001@E1_200
	E(□+1)_0 to E(□+1)_32767 in the bank following the specified bank @ are specified if @E□_□□□□□ contains 8000 to FFFF hex (32768 to 65535 decimal).	@E1_200 &32770 Contents Hex: #8002 Specifies E2_2.	

*2 When specifying an indirect address in Binary Mode, treat the DM Area and the EM Area (banks 0 to 18 hex) as one series of addresses. If the contents of an address with the @ symbol exceeds 32767, the address will be assumed to be an address in the EM Area continuing on from 0 in bank 0.

Example: If the DM Area word contains 32768, E0_0 in bank 0 in the EM Area would be specified.

If the EM Area bank number is specified as “n” and the contents of the word exceeds 32767, the address will be assumed to be an address in the EM Area continuing on from 0 in bank n+1.

Example: If bank 2 in the EM Area contains 32768, E3_0 in bank 3 in the EM Area would be specified.

Operand	Description	Notation	Application examples
Specifying indirect DM/EM addresses in BCD Mode	The offset from the beginning of the area is specified. The contents of the address will be treated as BCD data (0000 to 9999) to specify the word address in the DM Area or the EM Area. Add an asterisk (*) at the front to specify an indirect address in BCD Mode. 		MOV #0001 *D200

Operand	Description		Notation	Application examples
Specifying a register directly	An index register (IR) or a data register (DR) is specified directly by specifying IR□ (□: 0 to 15) or DR□ (□: 0 to 15).		IR0	MOVR 1.02 IR0 Stores the PLC memory address for CIO 0010 in IR0.
			IR1	MOVR 10 IR1 Stores the PLC memory address for CIO 0010 in IR1.
Specifying an indirect address using a register	Indirect address (No offset)	The bit or word with the PLC memory address contained in IR@ will be specified. Specify ,IR□ to specify bits and words for instruction operands.	,IR0	LD ,IR0 Loads the bit with the PLC memory address in IR0.
			,IR1	MOV #0001 ,IR1 Stores #0001 in the word with the PLC memory in IR1.
	Constant offset	The bit or word with the PLC memory address in IR□ + or – the constant is specified. Specify +/– constant ,IR□. Constant offsets range from –2048 to +2047 (decimal). The offset is converted to binary data when the instruction is executed.	+5 ,IR0	LD +5 ,IR0 Loads the bit with the PLC memory address in IR0 + 5.
			31 ,IR1	MOV #0001 +31 ,IR1 Stores #0001 in the word with the PLC memory address in IR1 + 31
	DR offset	The bit or word with the PLC memory address in IR□ + the contents of DR□ is specified. Specify DR□ ,IR□. DR (data register) contents are treated as signed-binary data. The contents of IR□ will be given a negative offset if the signed binary value is negative.	DR0 ,IR0	LD DR0 ,IR0 Loads the bit with the PLC memory address in IR0 + the value in DR0.
			DR0 ,IR1	MOV #0001 DR0 ,IR1 Stores #0001 in the word with the PLC memory address in IR1 + the value in DR0.
	Auto Increment	The contents of IR□ is incremented by +1 or +2 after referencing the value as an PLC memory address. +1: Specify ,IR□+ +2: Specify ,IR□ + +	,IR0++	LD ,IR0 ++ Increments the contents of IR0 by 2 after the bit with the PLC memory address in IR0 is loaded.
			,IR1+	MOV #0001 ,IR1 + Increments the contents of IR1 by 1 after #0001 is stored in the word with the PLC memory address in IR1.
	Auto Decrement	The contents of IR□ is decremented by –1 or –2 after referencing the value as an PLC memory address. –1: Specify ,–IR□ –2: Specify ,– –IR□	,--IR0	LD ,– –IR0 After decrementing the contents of IR0 by 2, the bit with the PLC memory address in IR0 is loaded.
			,IR1	MOV #0001 ,–IR1 After decrementing the contents of IR1 by 1, #0001 is stored in the word with the PLC memory address in IR1.

Data	Operand	Data form	Symbol	Range	Application example
16-bit constant	All binary data or a limited range of binary data	Unsigned binary	#	#0000 to #FFFF	MOV #0100 D0 Stores #0100 hex (&256 decimal) in D0. +#0009 #0001 D1 Stores #000A hex (&10 decimal) in D1.
		Signed decimal	±	–32768 to +32767	MOV –100 D0 Stores –100 decimal (#FF9C hex) in D0. +–9 -1 D1 Stores –10 decimal (#FFF6 hex) in D1.
		Unsigned decimal	&	&0 to &65535	MOV &256 D0 Stores –256 decimal (#0100 hex) in D0. +&9 &1 D1 Stores –10 decimal (#000A hex) in D1.
	All BCD data or a limited range of BCD data	BCD	#	#0000 to #9999	MOV #0100 D0 Stores #0100 (BCD) in D0. +B #0009 #0001 D1 Stores #0010 (BCD) in D1.
32-bit constant	All binary data or a limited range of binary data	Unsigned binary	#	#00000000 to #FFFFFFFF	MOVL #12345678 D0 Stores #12345678 hex in D0 and D1. D1 D0 1234 5678
		Signed binary	+	–2147483648 to +2147483647	MOVL –12345678 D0 Stores –12345678 decimal in D0 and D1.
		Unsigned decimal	&	&0 to &4294967295	MOVL &12345678 D0 Stores &12345678 decimal in D0 and D1.
	All BCD data or a limited range of BCD data	BCD	#	#00000000 to #99999999	MOVL #12345678 D0 Stores #12345678 (BCD) in D0 and D1

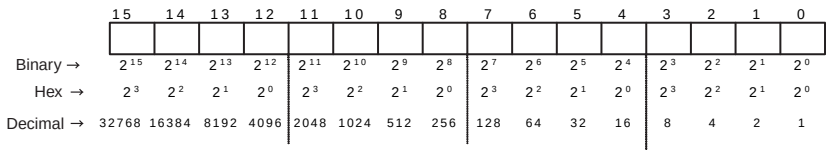
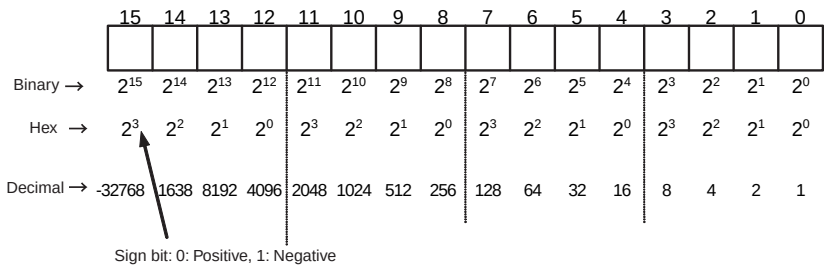
Data	Description	Symbol	Examples	---
Text string	Text string data is stored in ASCII (one byte except for special characters) in order from the leftmost to the rightmost byte and from the rightmost (smallest) to the leftmost word. 00 hex (NUL code) is stored in the rightmost byte of the last word if there is an odd number of characters. 0000 hex (2 NUL codes) is stored in the leftmost and rightmost vacant bytes of the last word + 1 if there is an even number of characters.	---	↓ ABCDE 'A' 'B' 'C' 'D' 'E' NUL = 41 42 43 44 45 00 ABCD 'A' 'B' 'C' 'D' NUL NUL = 41 42 43 44 00 00	MOV$ D100 D200 D100 41 42 D101 43 44 D102 45 00 ↓ D200 41 42 D201 43 44 D202 45 00

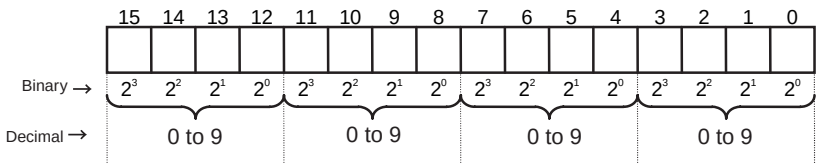
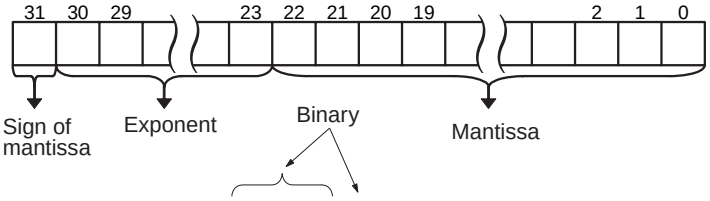
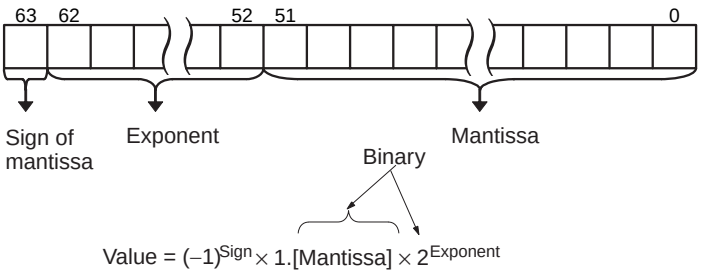
ASCII characters that can be used in a text string includes alphanumeric characters, Katakana and symbols (except for special characters). The characters are shown in the following table.

Lower four digits \ Upper four digits	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			sp	0	@	P	`	p				ー	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			″	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			$	4	D	T	d	t			、	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			′	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x			ィ	ク	ネ	リ		
9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C			,	<	L	¥	l	|			ャ	シ	フ	ワ		
D			−	=	M	]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	~			ョ	セ	ホ	゛		
F			/	?	O	_	o				ッ	ソ	マ	°		

5-6-3 Data Formats

The following table shows the data formats that the CJ Series can handle.

Data type	Data format	Decimal	4-digit hexa-decimal
Unsigned binary		&0 to &65535	#0000 to #FFFF
Signed binary	 The data is treated as signed binary 16-bit data using the leftmost bit as the sign bit. The value is expressed in 4-digit hexadecimal. Positive numbers: If the leftmost bit is OFF, it indicates a non-negative value. For 4-digit hexadecimal, the value will be 0000 to 7FFF hex. Negative numbers: If the leftmost bit is ON, it indicates a negative value. For 4-digit hexadecimal, the value be 8000 to FFFF hex and it will be expressed as the 2's complement of the absolute value of the negative value (decimal).	0 to –32768 0 to +32767	Negative: #8000 to #FFFF Positive: #0000 to #7FFF

Data type	Data format	Decimal	4-digit hexa-decimal
BCD (binary coded decimal)		#0 to #9999	#0000 to #9999
Single-precision floating-point decimal	 Value = $(-1)^{Sign} \times 1.[Mantissa] \times 2^{Exponent}$ Sign (bit 31): 1: negative or 0: positive Mantissa: The 23 bits from bit 00 to bit 22 contain the mantissa, i.e., the portion below the decimal point in 1.□□□□....., in binary. Exponent: The 8 bits from bit 23 to bit 30 contain the exponent. The exponent is expressed in binary as 127 plus n in 2^n. This format conforms to IEEE754 standards for single-precision floating-point data and is used only with instructions that convert or calculate floating-point data. It can be used to set or monitor from the I/O memory Edit and Monitor Screen on the CX-Programmer. As such, users do not need to know this format although they do need to know that the formatting takes up two words.	---	---
Double-precision floating-point decimal	 Value = $(-1)^{Sign} \times 1.[Mantissa] \times 2^{Exponent}$ Sign (bit 63): 1: negative or 0: positive Mantissa: The 52 bits from bit 00 to bit 51 contain the mantissa, i.e., the portion below the decimal point in 1.□□□□....., in binary. Exponent: The 11 bits from bit 52 to bit 62 contain the exponent The exponent is expressed in binary as 1023 plus n in 2^n. This format conforms to IEEE 754 standards for double-precision floating-point data and is used only with instructions that convert or calculate floating-point data. It can be used to set or monitor from the I/O memory Edit and Monitor Screen on the CX-Programmer. As such, users do not need to know this format although they do need to know that the formatting takes up four words.	---	---



Additional Information

- Complements
  Generally the complement of base x refers to a number produced when all digits of a given number are subtracted from x-1 and then 1 is added to the rightmost digit. (Example: The ten's complement of 7556 is 9999 – 7556 + 1 = 2444.) A complement is used to express a subtraction and other functions as an addition.

  Example: With 8954 – 7556 = 1398, 8954 + (the ten's complement of 7556) = 8954 + 2444 = 11398. If we ignore the leftmost bit, we get a subtraction result of 1398.

- Two's Complements
  A two's complement is a base-two complement. Here, we subtract all digits from 1 (2 – 1 = 1) and add one.

  Example: The two's complement of binary number 1101 is 1111 (F Hex) – 1101 (D Hex) + 1 (1 Hex) = 0011 (3 Hex). The following shows this value expressed in 4-digit hexadecimal.
  The two's complement b Hex of a Hex is FFFF Hex – a Hex + 0001 Hex = b Hex. To determine the two's complement b Hex of “a Hex,” use b Hex = 10000 Hex – a Hex.

  Example: to determine the two's complement of 3039 Hex, use 10000 Hex – 3039 Hex = CFC7 Hex.
  Similarly use a Hex = 10000 Hex – b Hex to determine the value a Hex from the two's complement b Hex.

  Example: To determine the real value from the two's complement CFC7 Hex use 10000 Hex – CFC7 Hex = 3039 Hex.
  The CJ Series has two instructions: NEG(160)(2'S COMPLEMENT) and NEGL(161) (DOUBLE 2'S COMPLEMENT) that can be used to determine the two's complement from the true number or to determine the true number from the two's complement.

● Values Represented in 1-word Data

Value (Decimal)	Binary representation			BCD representation (decimal)
	Decimal representations		Hexadecimal representation	
	Unsigned	Signed		
1	&1	+1	#0001	#0001
2	&2	+2	#0002	#0002
3	&3	+3	#0003	#0003
4	&4	+4	#0004	#0004
5	&5	+5	#0005	#0005
6	&6	+6	#0006	#0006
7	&7	+7	#0007	#0007
8	&8	+8	#0008	#0008
9	&9	+9	#0009	#0009
10	&10	+10	#000A	#0010
11	&11	+11	#000B	#0011
12	&12	+12	#000C	#0012
13	&13	+13	#000D	#0013
14	&14	+14	#000E	#0014
15	&15	+15	#000F	#0015
16	&16	+16	#0010	#0016
:	:	:	:	:
9999	&9999	+9999	#270F	#9999
10000	&10000	+10000	#2710	Not applicable.
	:	:	:	
32767	&32767	+32767	#7FFF	
32768	&32768	Not applicable.	#8000	
:	:		:	
65535	&65535		#FFFF	
–1	Not applicable.	–1	#FFFF	Not applicable.
:		:	:	
–32768		–32768	#8000	
–32769		Not applicable.	Not applicable.	

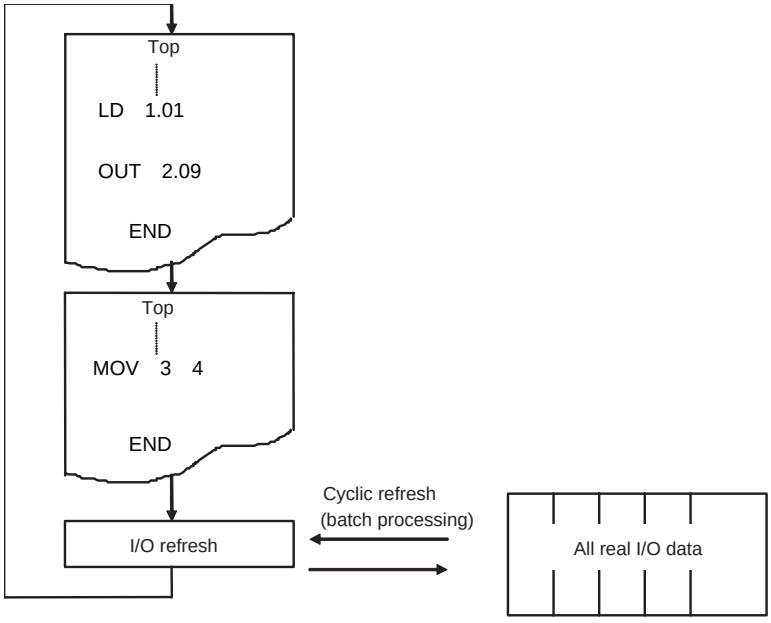
5-6-4 I/O Refresh Timing

The following methods are used to refresh external I/O.

- Cyclic refresh
- Immediate refresh (instruction with the ! specification, IORF(097), FIORF(225), or DLNK(226))

Cyclic Refresh

With cyclic refreshing, I/O refreshing is all performed at once after execution of all the cyclic tasks in READY status has been completed.



Execute an IORF(097) instruction for all required words prior to the END(001) instruction if I/O refreshing is required in a task.

Immediate Refresh

With immediate refreshing, refreshing is performed whenever an instruction is executed in the program.

● Types of Immediate Refresh

There are two ways to perform immediate refreshing: using an instruction and specify the immediate refresh variation (!) or using a special instruction for immediate refreshing.

Instructions with Refresh Variation (!)

Refreshing can be performed when the instruction is executed (i.e., during the cycle) by using the immediate refresh variation (!) of the instruction.

Special Instructions for Immediate Refresh

The following three instructions can be used depending on the Unit for which memory is to be refreshed.

IORF(097) instruction: Basic I/O Units and Special I/O Units (word allocated in CIO Area only)

FIORF(225) instruction: Special I/O Units

DLNK(226) instruction: CPU Bus Units

The following table shows which Units can be refreshed with which instructions.

Unit	Instruction with immediate refresh variation (!)	IORF(097) instruction	FIORF(225) instruction	DLNK(226) instruction
Basic I/O Units	Yes	Yes	No	No
Words allocated in CIO Area				
Special I/O Units	No	Yes*2	Yes*2	No
Words allocated in CIO Area				
Words allocated in DM Area		No		
CPU Bus Units	No	No	No	YES
Words allocated in CIO Area				
Words allocated in DM Area				
Unit-specific memory*1				

*1 EtherNet/IP Units, Controller Link Units, data links for SYSMAC LINK Units and DeviceNet remote I/O communications.

*2 Words allocated in the CIO Area to Special I/O Units can be refreshed by using either the IORF(097) instruction or the FIORF(225) instruction. Instruction execution time is shorter with the FIORF(225) instruction than with the IORF(097) instruction. It is also easier to use because all that is required is specifying the unit number of the Special I/O Unit in the operand.

Precautions for Correct Use

The execution times for immediate-refreshing variations are longer than the regular variations of instructions, so be careful because the cycle time will be longer.

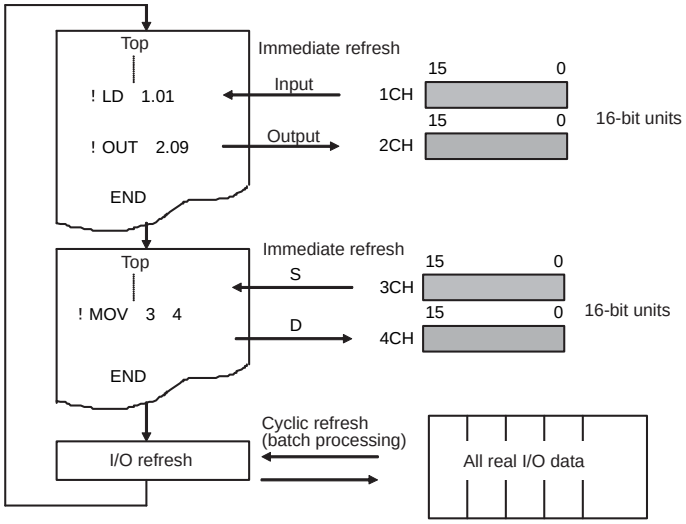
● Instructions with Refresh Variation (!)

Add an exclamation mark (!) in front of the instruction to specify immediate refreshing.

- I/O will be refreshed as shown below when an instruction is executing if an real I/O bit is specified as an operand.

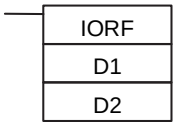
Units	Refreshed data
Basic I/O Units	I/O will be refreshed for the 16 bits containing the bit.

- When a word operand is specified for an instruction, I/O will be refreshed for the 16 bits that are specified.
- Inputs will be refreshed for input or source operand just before an instruction is executed.
- Outputs will be refreshed for outputs or destination (D) operands just after an instruction is execute.



● IORF(097): I/O REFRESH

IORF(097) can be used to refresh all data during a cycle for actual I/O in Basic I/O Units and data in CIO Area words allocated to Special I/O Units.

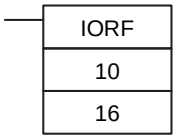


D1: Start word

D2: End word

The data in all words from D1 through D2 are refreshed when IORF(097) is executed.

Example:



The 7 words from CIO 10 through CIO 16 are refreshed when IORF(097) is executed.

When a high-speed response is needed from a calculation that uses input data from a Basic I/O Unit or outputs data to a Basic I/O Unit, use IORF(097) just before and just after the calculation instruction.

Precautions for Correct Use

IORF(097) has a relatively long instruction execution time and that execution time increases proportionally with the number of words being refreshed, so it can significantly increase the cycle time. Be careful not to let the cycle time become too long.

● FIORF(225): SPECIAL I/O UNIT I/O REFRESH

FIORF(225) can be used to refresh the following data in a Special I/O Unit with the specified unit number, only when necessary.



N: Special I/O Unit's unit number

Example:



Refreshes the CIO words and DM words allocated to the Special I/O Unit with unit number &20.

A Special I/O Unit's regular cyclic I/O refreshing can be disabled in the PLC Setup (by turning ON the Unit's Special I/O Unit Cyclic Refresh Disable Bit), and I/O refreshing can be performed with the Unit only when necessary by executing FIORF(225). This function can prevent the PLC's cycle time from increasing when a Special I/O Unit is connected in the PLC.

● Using DLNK(226): CPU Bus Unit I/O Refresh

DLNK(226) can be used to increase the refresh frequency for words allocated to a CPU Bus Unit in the CIO Area and DM Area, and for data that is specific to the CPU Bus Unit.*1

*1 Data specific to a CPU Bus Unit would include data links for Controller Link Unit or SYSMAC LINK Units, as well as remote I/O for DeviceNet Units.



N: Unit number of CPU Bus Unit

Example:



The instruction on the left would refresh the words allocated to the Unit in the CIO Area and DM Area, and data specific to the CPU Bus Unit*1 with unit number 1.

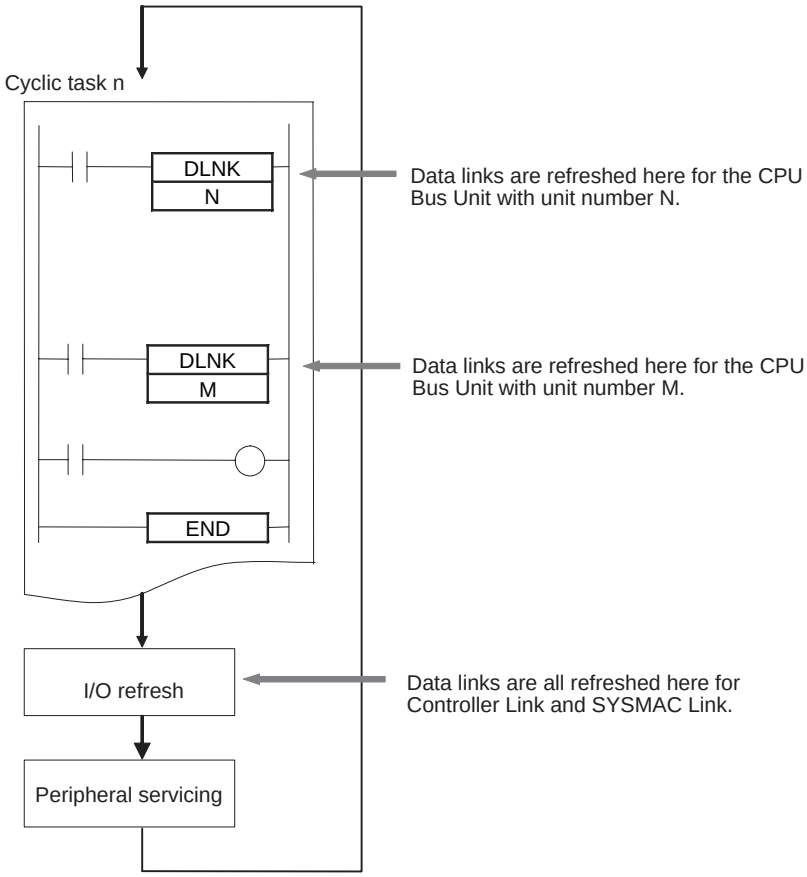
*1 If the CPU Bus Unit with unit number 1 is a Controller Link Unit, data links are refreshed.

Application Example: With a long cycle time, the refresh interval for Controller Link data links can be very long. This interval can be shortened by executing DLNK(226) for the Controller Link Unit to increase the frequency of data link refreshing.

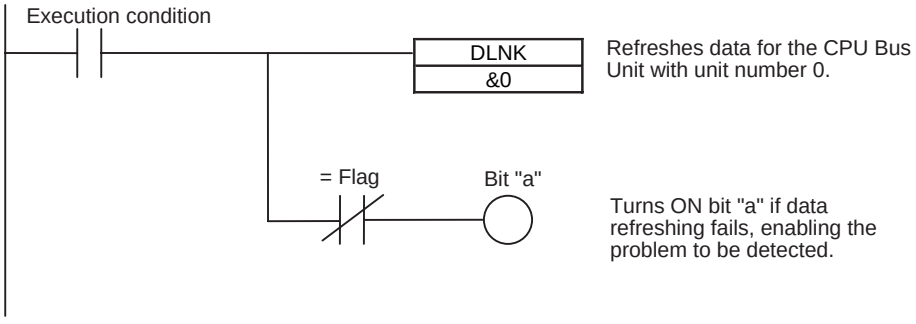
Additional Information

The following figure shows an example application of a DLNK(226) instruction.

Example: Data Links



If DLNK(226) is executed for a CPU Bus Unit that is busy refreshing data, data will not be refreshed and the Equals Flag will turn OFF. Normally, the Equals Flag should be programmed as shown below to be sure that refreshing has been completed normally.

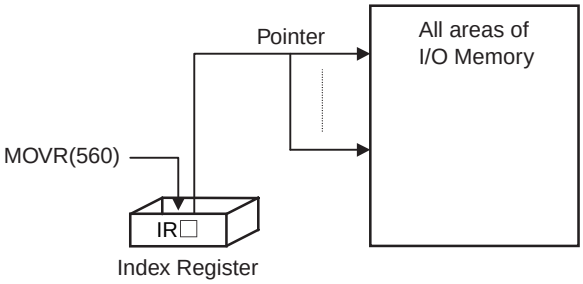


5-7 Index Registers

5-7-1 What Are Index Registers?

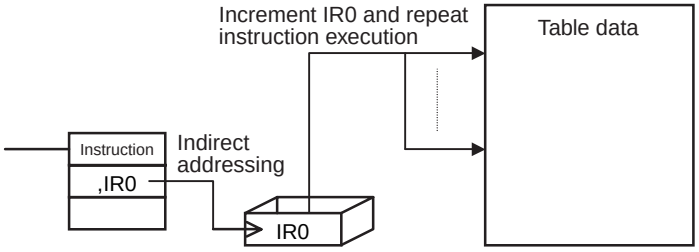
Index Registers function as pointers to specify PLC memory addresses, which are absolute memory addresses in I/O memory. After storing a PLC memory address in an Index Register with MOVR(560) or MOVRW(561), input the Index Register as an operand in other instructions to indirectly address the stored PLC memory address.

The advantage of Index Registers is that they can specify any bit or word in I/O memory, including timer and counter PVs.



5-7-2 Using Index Registers

Index Registers can be a powerful tool when combined with loops such as FOR-NEXT loops. The contents of Index Registers can be incremented, decremented, and offset very easily, so a few instructions in a loop can process tables of consecutive data very efficiently.



Basically, Index Registers are used with the following steps:

1 Use MOVR(560) to store the PLC memory address of the desired bit or word in an Index Register.

2 Specify the Index Register as the operand in almost any instruction to indirectly address the desired bit or word.

3 Offset or increment the original PLC memory address (see below) to redirect the pointer to another address.

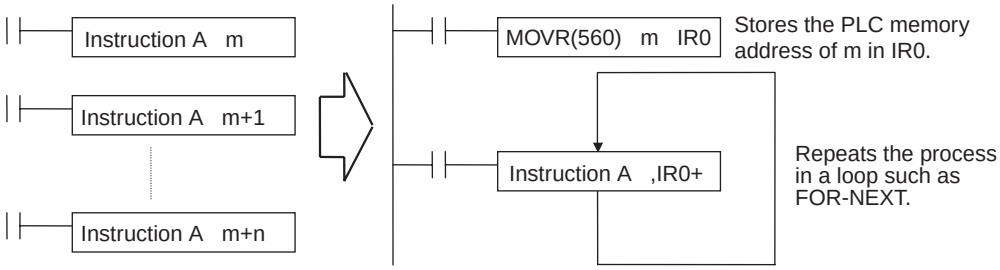
4 Continue steps 2 and 3 to execute the instruction on any number of addresses.

Note Use any of the following methods to offset or increment an Index Register.

- Indirect Addressing of Index Registers:
  Indirect addressing with auto-incrementing (,IR□+ or ,IR□++), indirect addressing with auto-decrementing (,–IR□ or ,––IR□), indirect referencing with a constant offset (constant,IR□), indirect addressing with a DR offset (DR□,IR□)

- Instructions for Direct Addressing of Index Registers:
  BINARY ADD (+L), BINARY SUBTRACT (–L), DOUBLE INCREMENT BINARY (++L), DOUBLE DECREMENT BINARY (––L)

Example:



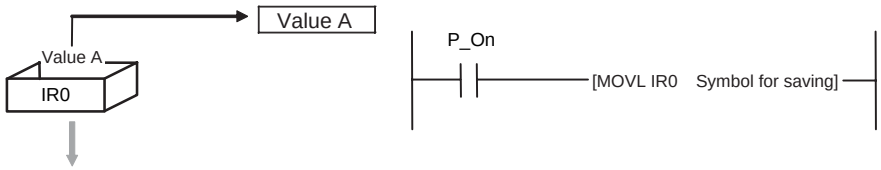
The following example shows how an Index Register in a program loop can replace a long series of instructions. In this case, instruction A is repeated n+1 times to perform some operation such as reading and comparing a table of values.

Precautions for Correct Use

- When Index Registers IR0 to IR15 are used within function blocks, using the same Index Register within other function blocks or in the program outside of function blocks will create competition between the two instances and the program will not execute properly. Therefore, when using Index Registers (IR0 to IR15), always save the value of the Index Register at the point when the function block starts (or before the Index Register is used), and when the function block is completed (or after the Index Register has been used), incorporate processing in the program to return the Index Register to the saved value.

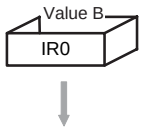
  Example: Starting function block (or before using Index Register):

  1. Save the value of IR (e.g., A).



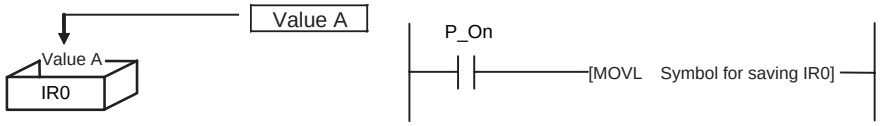
  Within function block:

  2. Use IR.



  At start of function block (or before Index Register is used):

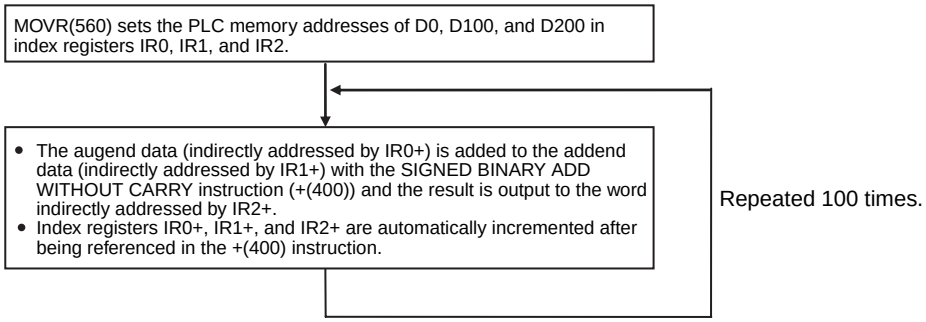
  3. Return IR to saved value (e.g., A).

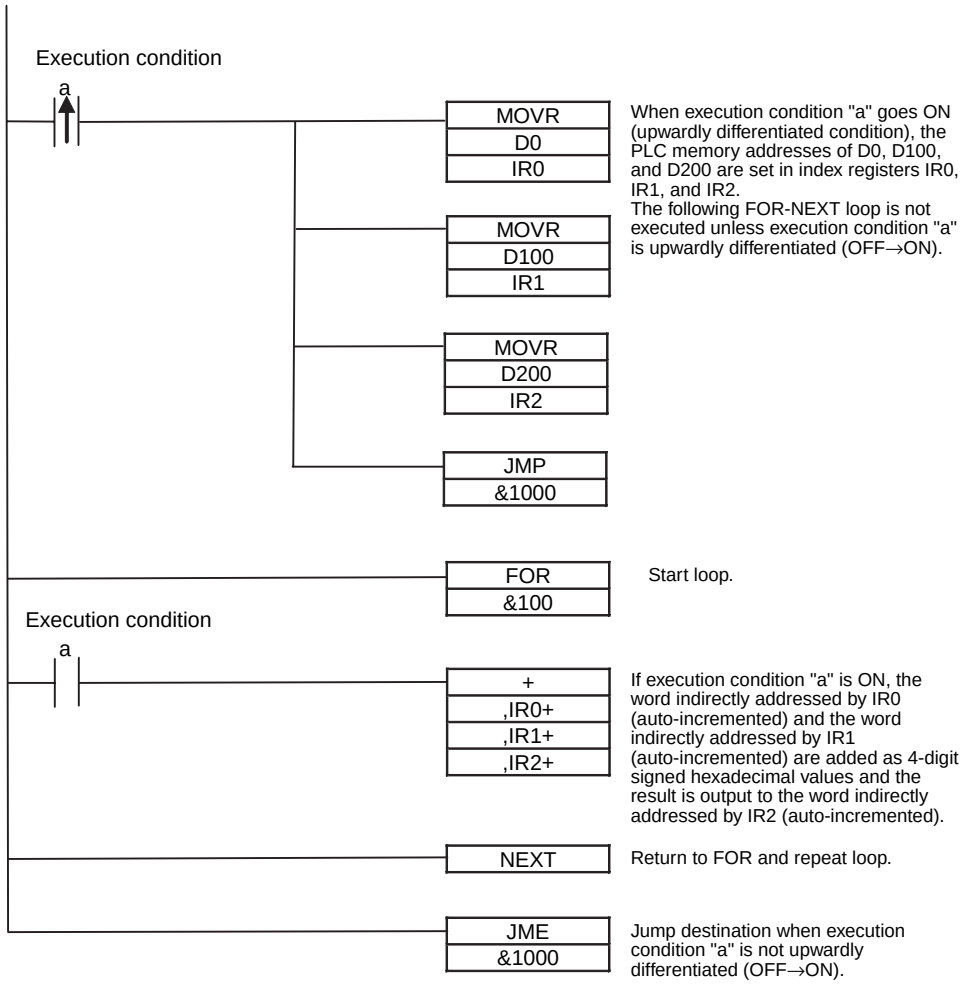


- Always set the value before using Index Registers. Operation will not be stable if Index Registers are used without the values being set.

Application Example for Index Registers

The data in D0 to D99 (augend data) is added to the data in D100 to D199 (addend data) and the addition results are output to D200 to D299. The operands of a single addition instruction are specified by index registers and the addition operations are performed by incrementing the index registers and repeatedly executing the addition instruction.





Additional Information

Index Registers can be directly addressed only in the instructions shown in the following table.

Direct Addressing of Index Registers

The size of an index registers is two words per register for Index Registers IR0 to IR15, so use a double-word instruction (with an “L” in the mnemonic).

Instruction group	Instruction name	Mnemonic	Primary function
Data Movement Instructions	MOVE TO REGISTER	MOVR(560)	Stores the PLC memory address of a bit or word in an Index Register.
	MOVE TIMER/COUNTER PV TO REGISTER	MOVRW(561)	
Table Data Processing Instructions	SET RECORD LOCATION	SETR(635)	
	GET RECORD NUMBER	GETR(636)	Outputs the PLC memory address stored in an Index Register.
Tracking Instructions	Record Search Instructions	RSRCH□ (Function codes: 360 to 364, 370 to 374, and 380 to 384)	Outputs the first PLC memory address of the matching record to an Index Register.
	Record Sort Instructions	RSORT□ (Function codes: 203 205)	Outputs one higher than the PLC memory address of the last address in the sorted range to an Index Register.
Data Movement Instructions	DOUBLE MOVE	MOVL(498)	Transfers between Index Registers. Used for exchanges and comparisons.
	DOUBLE DATA EXCHANGE	XCGL(562)	
Comparison Instructions	DOUBLE EQUAL	=L(301)	
	DOUBLE NOT EQUAL	< >L(306)	
	DOUBLE LESS THAN	< L(311)	
	DOUBLE LESS THAN OR EQUAL	< =L(316)	
	DOUBLE GREATER THAN	>L(321)	
	DOUBLE GREATER THAN OR EQUAL	>=L(326)	
	DOUBLE COMPARE	CMPL(060)	
Increment/Decrement Instructions	DOUBLE INCREMENT BINARY	++L(591)	Changes the PLC memory address in the Index Register by incrementing, decrementing, or offsetting its content.
	DOUBLE DECREMENT BINARY	–-L(593)	
Symbol Math Instructions	DOUBLE SIGNED BINARY ADD WITHOUT CARRY	+L(401)	
	DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY	–L(411)	
Special Instructions	CONVERT ADDRESS FROM CV	FRMCV(284)	Convert actual PLC memory addresses between CVM1/CV-series and CJ-series addresses.
	CONVERT ADDRESS TO CV	TOCV(285)	

5-7-3 Processing Related to Index Registers

The CJ-series CPU Unit's Table Data Processing Instructions complement the functions of the Index Registers. Table Data Processing Instructions can be broadly divided into stack processing instructions and table processing instructions.

Processing			Purpose	Instructions
Stack processing			Operate FIFO (first-in first-out) or LIFO (last-in first-out) data tables, and read, write, insert, delete, or count data entries in data tables.	SSET(630), PUSH(632), FIFO(633), LIFO(634), SREAD(639), SWRIT(640), SINS(641), SDEL(642), SNUM(638)
Table processing	Tables with one-word records (Range instructions)	Basic processing	Find integer or floating-point decimal values such as for the maximum value, minimum value, or checksum.	MIN(183), MAX(182), MINL(175), MAXL(174), MINF(177), MAXDF(176), MIND(179), MAXD(178), SRCH(181), SUM(184), and FCS(180)
		Special processing	Perform various other table processing such as comparisons or sorting.	Combine Index Registers with instructions such as SRCH(181), MAX(182), MIN(183), and comparison instructions.
	Tables with multiple-word records		Search and sort records in a specified area (number of words and range) registered by the user.	RSRCH <(360), RSRCH <=(361), RSRCH =,(362), RSRCH >=(363), RSRCH >(364), RSRCH2 <(370), RSRCH2 <=(371), RSRCH2 =(372), RSRCH2 >=(373), RSRCH2 >(374), RSRCH4 <(380), RSRCH4 <=(381), RSRCH4 =(382), RSRCH4 >=(383), RSRCH4 >(384), RSORT(204), RSORT2(204), and RSORT4(205)
			Perform operations on records in a specified area (number of words and range) registered by the user.	Combine the following instructions with Index Registers. • DIM(631), SETR(635), and GETR(636) instructions • Other instructions (e.g., comparison instructions)

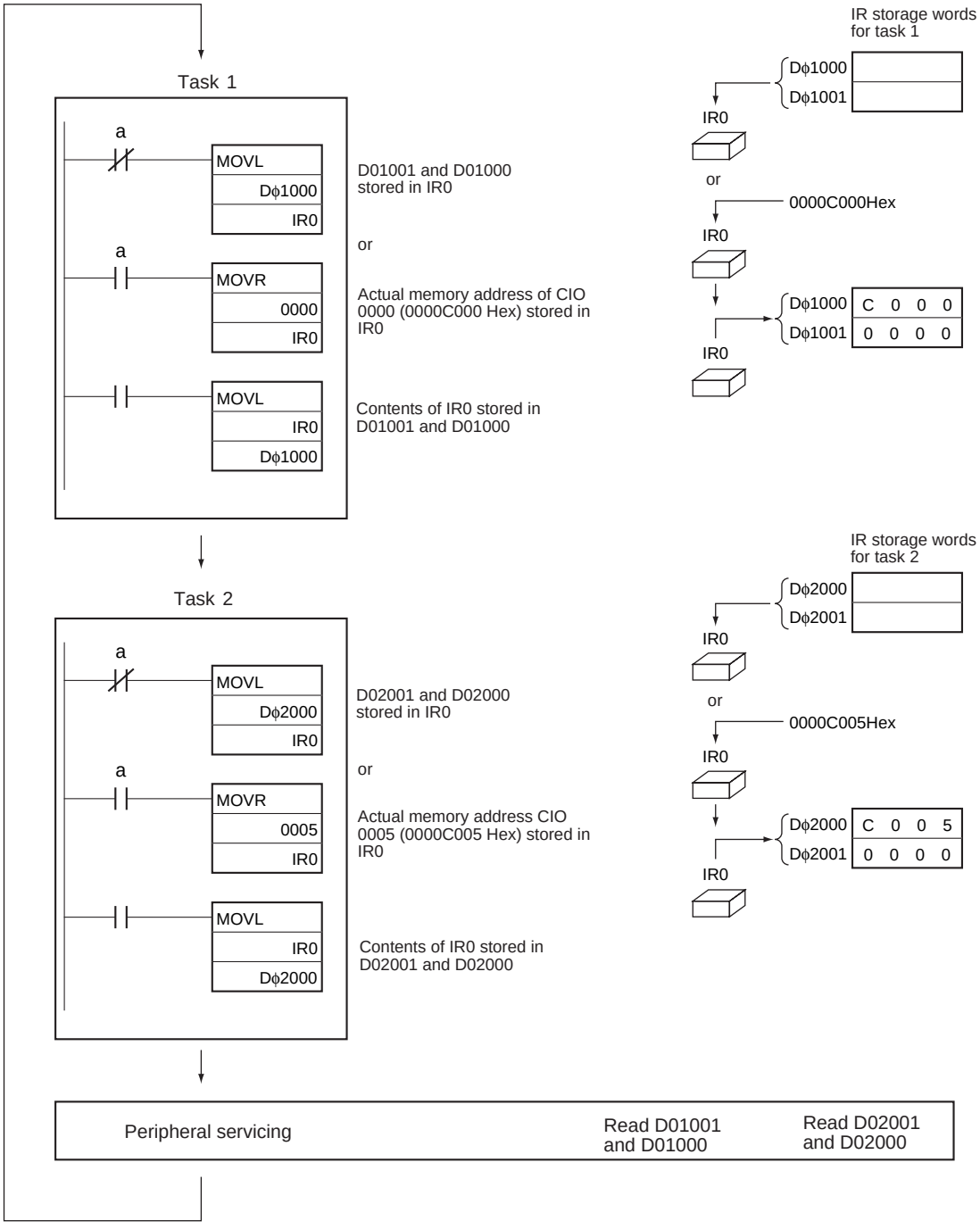
5-7-4 Monitoring Index Registers

It is possible to monitor Index Registers as follows:

- To use the CX-Programmer to monitor the final Index Register values for each task.
- To monitor the Index Register values using Host Link commands or FINS commands, write a program to store Index Register values from each task to another area (e.g., DM area) at the end of each task, and to read Index Register values from the storage words (e.g., DM area) at the beginning of each task. The values stored for each task in other areas (e.g., DM area) can then be edited using the CX-Programmer, Host Link commands, or FINS commands.

● Example

Note Be sure to use PLC memory addresses in Index Registers.



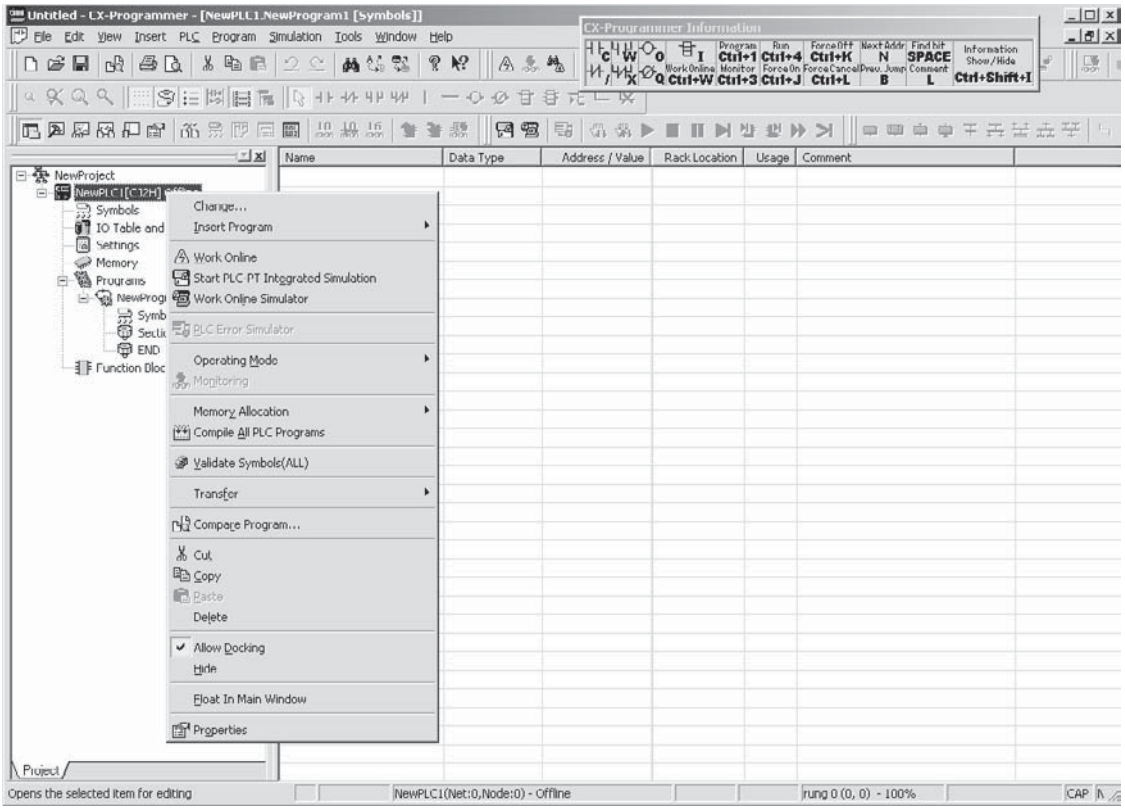
5-7-5 Sharing Index and Data Registers between Tasks

Index and Data Registers (IR/DR) can be shared between tasks. The normal setting is for separate registers for each task. The current setting can be confirmed in A99.14.

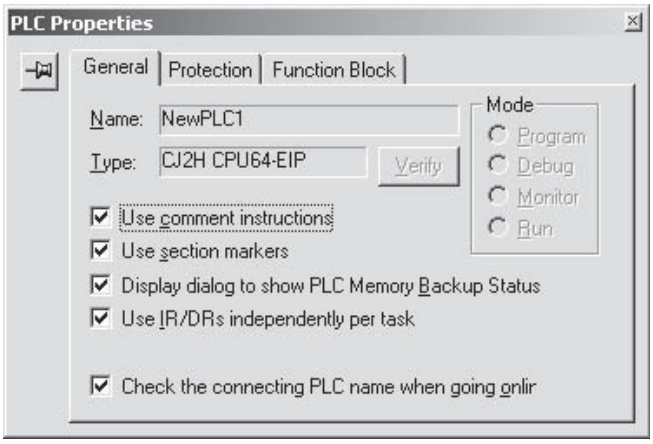
● Setting Method

Use the CX-Programmer to set shared index and data registers. This setting cannot be made from a Programming Console.

1 Select a PLC in the CX-Programmer project tree and click the right mouse button.



2 Select ***Properties***. The following PLC Properties Dialog Box will be displayed.



Leave the check mark for using IR/DR independently per task if separate index and data registers are required for each task. Remove the check mark to use shared index and data registers for all tasks.

● Auxiliary Area Flags and Words

Name	Address	Description
IR/DR Operation between Tasks	A99.14	Indicates whether or not index and data registers are shared between tasks. 0: Separate registers for each task (default) 1: Shared registers for all tasks



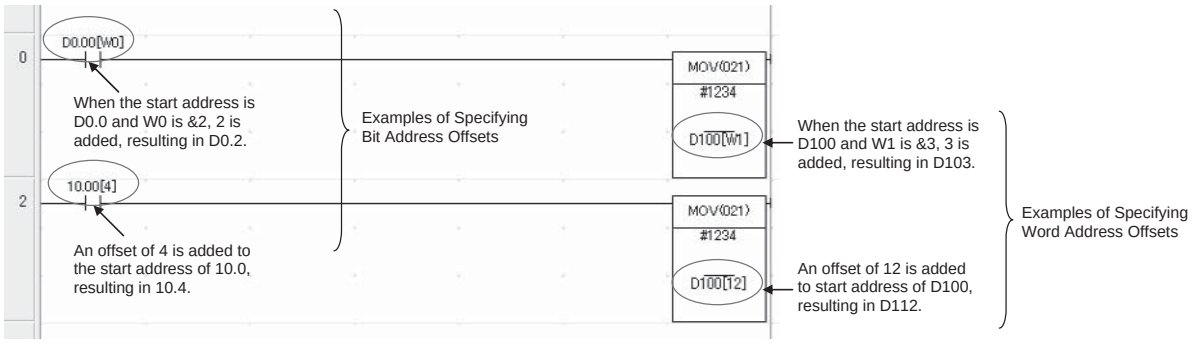
Additional Information

- Shared Index and Data Registers can be used to eliminate the need to store and load register contents between tasks when the same contents is needed in two or more tasks. Refer to *6-18 Index Registers* for information on storing and loading index register contents.
- The switching time between tasks will be somewhat faster when index and data registers are shared. It is recommended to set shared registers if the registers are not being used or if there is no particular need for separate registers in each task.

5-8 Specifying Address Offsets

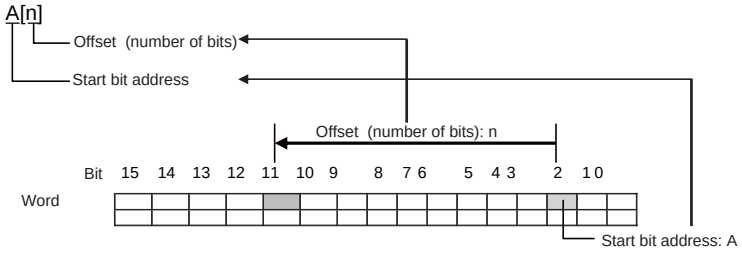
5-8-1 Overview

When an address is specified in an instruction operand, it is possible to change the specified address by specifying an offset for in brackets after the specified address.



● Bit Addresses

The bit address is offset by the amount specified by n (number of bits) from A (start bit address).



Start Bit Address

It is possible to specify the start bit address with a bit address or with a symbol (except for STRING or NUMBER data types). Offsetting is possible only for addresses in the H, W, DM, and EM Areas. I/O comments indicate the I/O comments for this start bit address.

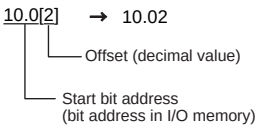
Offset

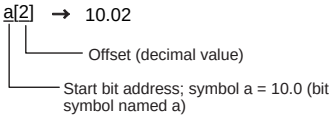
The offset can be specified as a decimal constant, word address (but CIO Area addresses cannot be specified), or one-word symbol (i.e., symbols with the following data types: INT, UINT, WORD, CHANNEL). If a word address is specified, the contents of the specified word is used as the offset.

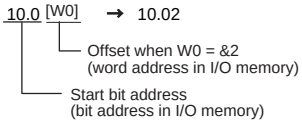
If the offset exceeds bit 15 in the specified word, offsetting will continue from bit 00 in the next word. If the offset is specified indirectly, make sure that the final bit address does not exceed the upper limit by using input comparison or other instruction.

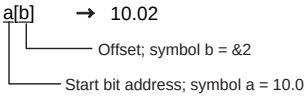
If the number of offset bits exceeds the memory area of the start bit address, the final bit address will be in the next memory area in the order determined by the actual PLC memory addresses. (For information on the arrangement of area in the memory map, refer to *A-4 Memory Map of PLC Memory Addresses*.

Examples:



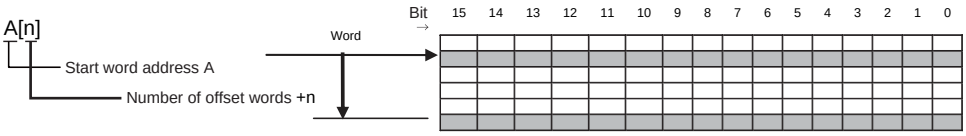






● Word Addresses

The word address is offset by the amount specified by n (number of offset words) from A (start word address).



Start Word Address

It is possible to specify the start words address with a word address or with a symbol (except for STRING or NUMBER data types). Offsetting is possible only for addresses in the H, W, DM, and EM Areas. I/O comments indicate the I/O comments for this start word address.

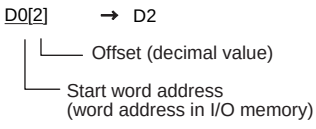
Offset

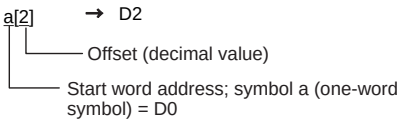
The offset can be specified as a decimal constant, word address (but CIO Area addresses cannot be specified), or one-word symbol (i.e., symbols with the following data types: INT, UINT, WORD, CHANNEL). If a word address is specified, the contents of the specified word is used as the offset.

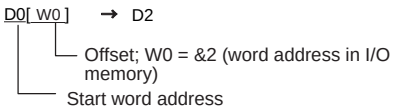
If the offset is specified indirectly, make sure that the final bit address does not exceed the upper limit by using input comparison or other instruction.

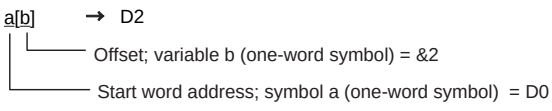
If the number of offset words exceeds the memory area of the start word address, the final word address will be in the next memory area in the order determined by the actual PLC memory addresses. (For information on the arrangement of area in the memory map, refer to *A-4 Memory Map of PLC Memory Addresses*.

Examples:









⚠ Caution

Program so that the memory area of the start address is not exceeded when a symbol or address is used to specify the offset directly in a ladder program.

If an indirect specification causes the address to exceed the memory area of the start address, the system will access data in the next area, and unexpected operation may occur.

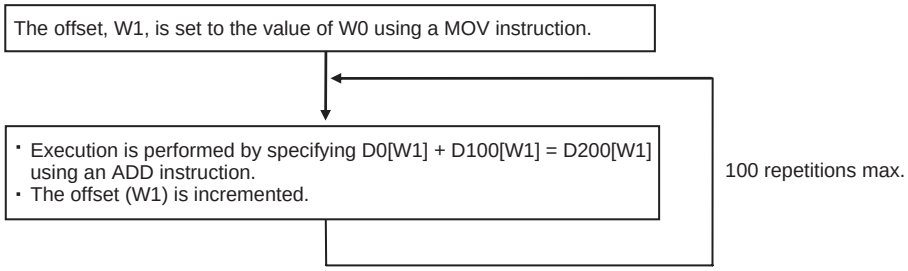


5-8-2 Examples of Address Offset Application

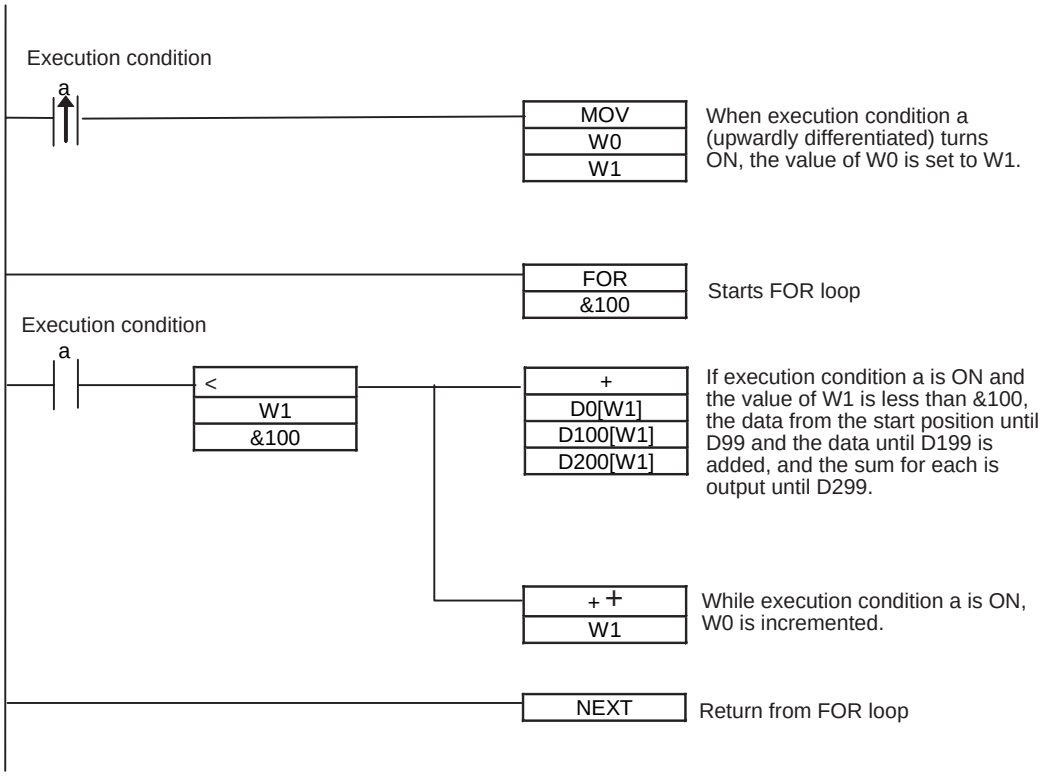
It is possible to dynamically specify the offset by specifying a word address in I/O memory for the offset in the brackets. The contents of the specified word address will be used as the offset. For example, execution can be performed by increasing the address by incrementing the value in the brackets and using only one instruction.

● Example of Ladder Programming

In this example, two areas of consecutive data are used: D0 to D99 and D100 to D199. The contents of corresponding words are added starting from the specified starting point, W0, to the end of the areas and the sums are output to D200 to D299 starting from the specified offset from D200. For example, if W0 is 30, the corresponding words from D30 to D99 and D130 to D199 are added, and output the sums are output to D230 to D299.



Each process is performed with an input comparison instruction (<) as the execution condition so that W1 does not exceed &100 to make sure that the upper limit of the indirect addressing range is not exceeded.



5-9 Checking Programs

CJ-series programs can be checked at the following four stages.

- Input check during CX-Programmer input operations
- Program check by CX-Programmer
- Instruction check during execution
- Fatal error check (program errors) during execution

5-9-1 Errors during CX-Programmer Input

The program will be automatically checked by the CX-Programmer at the following times.

Timing	Checked
When inputting ladder diagrams	Instruction inputs, operand inputs, programming patterns
When loading files	All operands for all instructions and all programming patterns
When downloading files	Models supported by the CJ Series and all operands for all instructions
During online editing	Capacity, etc.

The results of checking are output to the text tab of the Output Window. Also, the left bus bar of illegal program sections will be displayed in red in ladder view.

5-9-2 Program Checks with the CX-Programmer

The user program can be checked in the CX-Programmer. When the program is checked, the user can specify program check in any of four levels: A, B, or C (in order of the seriousness of the errors) or a custom check level.

The CX-Programmer does not check range errors for indirectly addressed operands in instructions. If an instruction's operand data is invalid, the ER Flag will be turned ON during the program execution check, which is described in the next section. For details, refer to the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474).

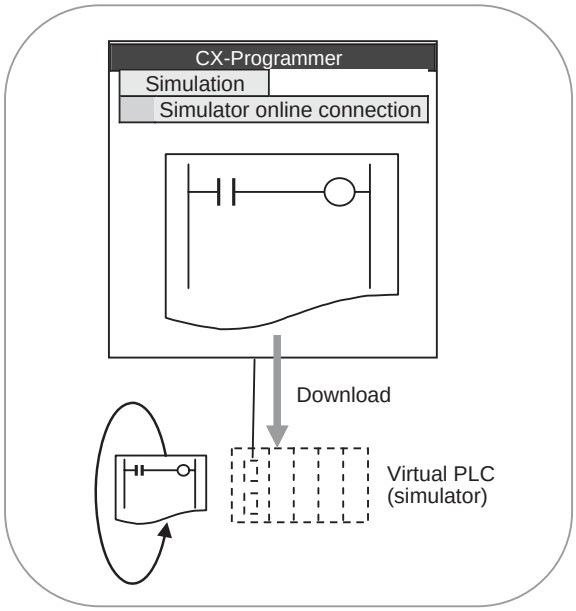
For details on the CX-Programmer's checks, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).

5-9-3 Debugging with the Simulator

Programming can be debugged without connecting to the actual PLC by simulating CPU Unit operation on a computer.

Checking Ladder Program Operation

Programming that has been created can be checked and debugged with a virtual PLC by starting the simulator in the CX-Simulator from the CX-Programmer.

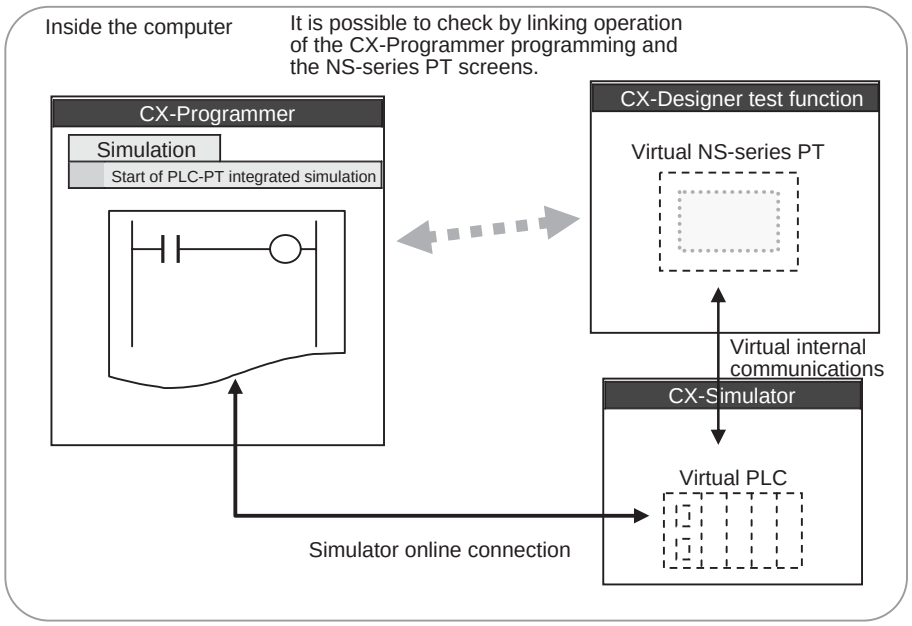


In addition to transferring programs and monitoring, the following functions can be used with the simulator. For details on the debugging procedure, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).

- Executing Step Run, Continuous Step Run, or Scan Run.
- Specifying break points, start points, and I/O break conditions.
- Checking the number of executions and execution time for each task.
- Simulating execution of interrupt tasks.
- Force-setting and force-resetting bits.

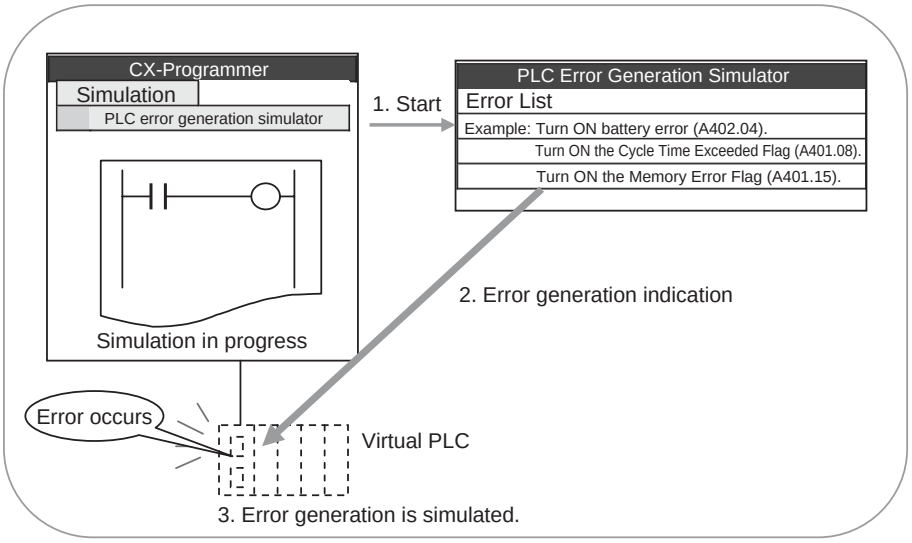
Debugging with Operation between PT and Ladder Programming: Integrated Simulation

With PLC-PT integrated simulation, it is possible to debug operation between screen data of an NS-series PT and a CJ-series PLC program. This is achieved by linking the simulator function of the CX-programmer and the offline test function of the CX-Designer PT screen design software. This enables debugging screens and screen controls from ladder programming using only a computer rather than connecting a computer and a PT with a cable. PLC-PT integrated simulation can also be started from the CX-Programmer. For information on the debugging procedure, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).



Error Simulation Function

With the CX-Programmer, it is possible to generate system errors in the virtual PLC during ladder programming simulation. It is easy to check operation of the ladder programming an the NS-series PT when a PLC system error occurs by generating the desired fatal or non-fatal system error using a special operation window. For information on the debugging procedure, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).



Note Unlike with an actual error, ladder execution will not stop even if a fatal error is generated using the PLC error generation simulation function.

Additional Information

System errors can also be generated in the PLC by using a FAL(006) or FALS(007) instruction.

5-9-4 Program Execution Check

The following checks can be performed using the CX-Programmer when programming is being executed.

The following four checks are performed during instruction execution.

Type of error	Flag that turns ON for error	Stop/Continue operation
Instruction Processing Error	ER Flag The Instruction Processing Error Flag (A295.08) will also turn ON if Stop Operation is specified when an error occurs.	A setting in the PLC Setup can be used to specify whether to stop or continue operation for instruction processing errors. The default is to continue operation. A program error will be generated and operation will stop only if Stop Operation is specified.
Access Error	AER Flag The Access Error Flag (A295.10) will turn ON if Stop Operation is specified when an error occurs.	A setting in the PLC Setup can be used to specify whether to stop or continue operation for instruction processing errors. The default is to continue operation. A program error will be generated and operation will stop only if Stop Operation is specified.
Illegal Instruction Error	Illegal Instruction Error Flag (A295.14)	Fatal (program error)
User Program Area Overflow Error	User Program Area Overflow Error Flag (A295.15)	Fatal (program error)

● Instruction Processing Errors (P_ER Flag ON Errors)

- An instruction processing error will occur if incorrect data was provided when executing an instruction or an attempt was made to execute an instruction outside of a task. Here, data required at the beginning of instruction processing was checked and as a result, the instruction was not executed, the P_ER Flag (Error Flag) will be turned ON and the P_EQ and P_N Flags may be retained or turned OFF depending upon the instruction.
  The P_ER Flag (error Flag) will turn OFF if the instruction (excluding input instructions) ends normally. Conditions that turn ON the P_ER Flag will vary with individual instructions. See descriptions of individual instructions in the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474) for more details.
- If Instruction Errors are set to Stop Operation in the PLC Setup, then operation will stop (fatal error) and the Instruction Processing Error Flag (A295.08) will turn ON if an instruction processing error occurs and the P_ER Flag turns ON.

● Illegal Access Errors (P_AER Flag ON Errors)

- Illegal access errors indicate that the wrong area was accessed in one of the following ways when the address specifying the instruction operand was accessed.
  - A read or write was executed for a parameter area.
  - A write was executed to a memory area that is not mounted*1
  - A write was executed to an EM Area bank specified as EM File Memory.
  - A write was executed to a read-only area.
  - The value specified in an indirect DM/EM address in BCD mode was not BCD (e.g., *D1 contains #A000).
- Instruction processing will continue and the Error Flag (ER Flag) will not turn ON if an access error occurs, but the Access Error Flag (P_AER Flag) will turn ON.

*1 An access error will occur for the following:
- When a specified EM address exceeds 32767 (example: E32768) for the current bank.
- The final bank is specified for an indirect EM address in BIN mode and the specified word contains 8000 to FFFF Hex (example: @EC_00001 contains #8000).

- The current bank is specified for an indirect EM address in BIN mode and the specified words contains 8000 to FFFF Hex (example: @EC_00001 contains #8000)
- An IR register containing the internal memory address of a bit is used as a word address or an IR containing the internal memory address of a word is used as a bit address.

- If Instruction Errors are set to Stop Operation in the PLC Setup, then operation will stop (fatal error) and the "Illegal Access Error Flag" (A295.10) will turn ON if an illegal access error occurs and the AER Flag turns ON.



Additional Information

The Access Error Flag (P_AER Flag) will not be cleared after a task is executed. If Instruction Errors are set to Continue Operation, this Flag can be monitored until just before the END(001) instruction to see if an illegal access error has occurred in the task program. (The status of the final P_AER Flag after the entire user program has been executed will be monitored if the AER Flag is monitored on the CX-Programmer.)

● Other Errors

Illegal Instruction Errors

Illegal instruction errors indicate that an attempt was made to execute instruction data other than that defined in the system. This error will normally not occur as long as the program is created with CX-Programmer.

In the rare even that this error does occur, it will be treated as a program error, operation will stop (fatal error), and the Illegal Instruction Flag (A295.14) will turn ON.

User Program Area Overflow Errors

User program area overflow errors indicate that an attempt was made to execute instruction data stored beyond the last address in the user program area defined as program storage area. This error will normally not occur as long as the program is created with CX-Programmer.

In the rare even that this error does occur, it will be treated as a program error, operation will stop (fatal error), and the UM Overflow Flag (A295.15) will turn ON.



Additional Information

If the Error Flag (P_ER) or Illegal Access Error Flag (P_AER) turns ON, it will be treated as a program error and it can be used to stop the CPU Unit from running. Specify operation for program errors in the PLC Setup.

● Program Errors

Program error	Description	Related flags
No END Instruction	An END instruction is not present in the program.	The No END Flag (A295.11) turns ON.
Error During Task Execution	No task is ready in the cycle. No program is allocated to a task. The corresponding interrupt task number is not present even though the execution condition for the interrupt task was met.	The Task Error Flag (295.12) turns ON.
Instruction Processing Error (P_ER Flag ON) and Stop Operation set for Instruction Errors in PLC Setup	The wrong data values were provided in the operand when an attempt was made to execute an instruction.	The ER Flag turns ON and the Instruction Processing Error Flag (A295.08) turns ON if Stop Operation set for Instruction Errors in PLC Setup.

Program error	Description	Related flags
Illegal Access Error (P_AER Flag ON) and Stop Operation set for Instruction Errors in PLC Setup	A read or write was executed for a parameter area. A read or write was executed for a memory area that is not mounted. A read or write was executed for an EM Area Bank specified as EM File Memory. A write was executed to a read-only area. The value specified in an indirect DM/EM address in BCD mode was not BCD.	AER Flag turns ON and the Illegal Access Error Flag (A295.10) turns ON if Stop Operation set for Instruction Errors in PLC Setup
Indirect DM/EM BCD Error and Stop Operation set for Instruction Errors in PLC Setup	The value specified in an indirect DM/EM address in BCD mode is not BCD.	The Access Error Flag will turn ON. If the Stop CPU on Instruction Error Check Box is selected in the PLC Setup, then the Indirect DM/EM BCD Error Flag (A295.09) will also turn ON.
Differentiation Address Overflow Error	During online editing, more than 131,071 differentiated instructions have been inserted or deleted.	The Differentiation Overflow Error Flag (A295.13) turns ON.
Illegal Instruction Error	An attempt was made to execute an instruction that cannot be executed.	The UM (User Memory) Overflow Flag (A295.14) turns ON.
UM (User Memory) Overflow Error	An attempt was made to execute instruction data stored beyond the last address in user memory (UM) defined as program storage area.	The UM (User Memory) Overflow Flag (A295.15) turns ON.

5-10 Precautions

5-10-1 Condition Flags

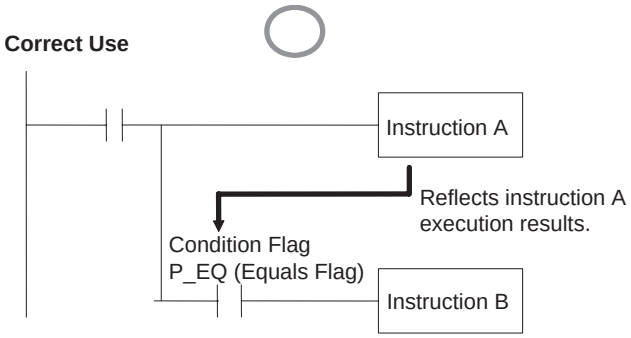
Using Condition Flags

Conditions flags are shared by all instructions, and will change during a cycle depending on results of executing individual instructions. Therefore, be sure to use Condition Flags on a branched output with the same execution condition immediately after an instruction to reflect the results of instruction execution. Never connect a Condition Flag directly to the bus bar because this will cause it to reflect execution results for other instructions.

Example: Using Instruction A Execution Results

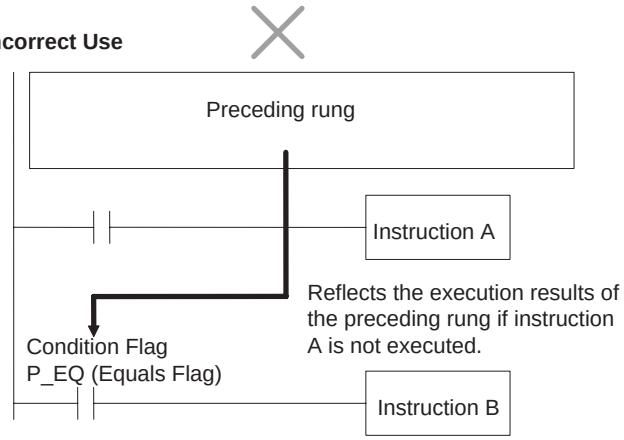
Mnemonic

Correct Use



The same execution condition (a) is used for instructions A and B to execute instruction B based on the execution results of instruction A. In this case, instruction B will be executed according to the Condition Flag only if instruction A is executed.

Incorrect Use

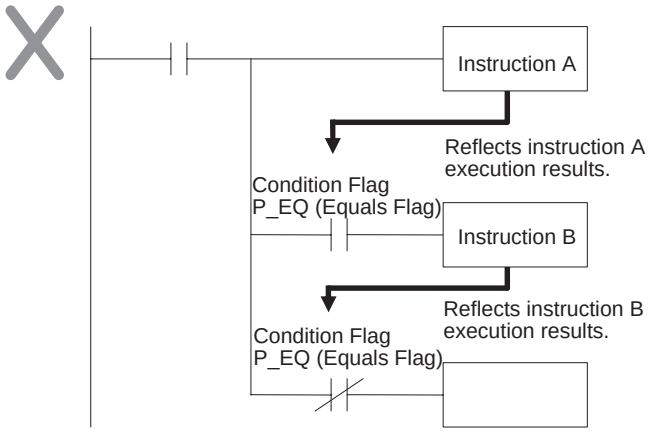


If the Condition Flag is connected directly to the left bus bar, instruction B will be executed based on the execution results of a previous rung if instruction A is not executed.

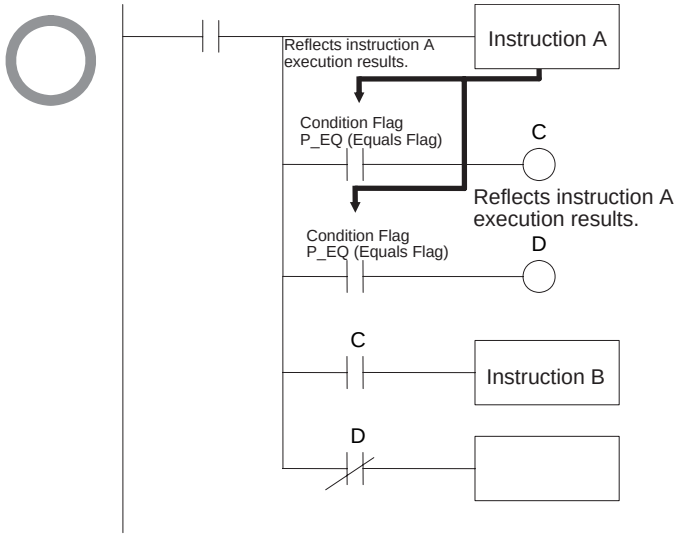
When interrupt tasks are being used, an interrupt task will operate when its start conditions are met, even during execution of a cyclic task. In this case, the Condition Flags are returned to their original status when processing switches back from the interrupt task to the cyclic task, even if those flags were turned ON/OFF in the interrupt task.

Using Execution Results in N.C. and N.O. Inputs

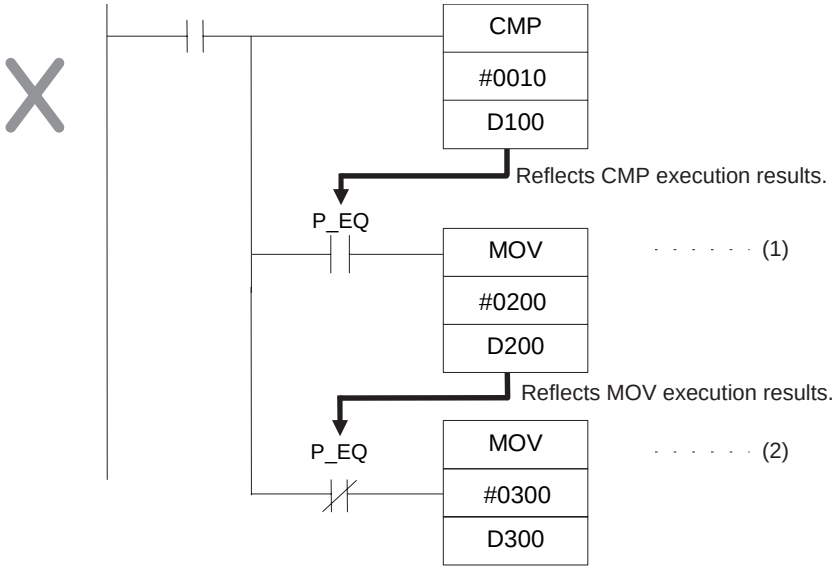
The Condition Flags will pick up instruction B execution results as shown in the example below even though the N.C. and N.O. input bits are executed from the same output branch.



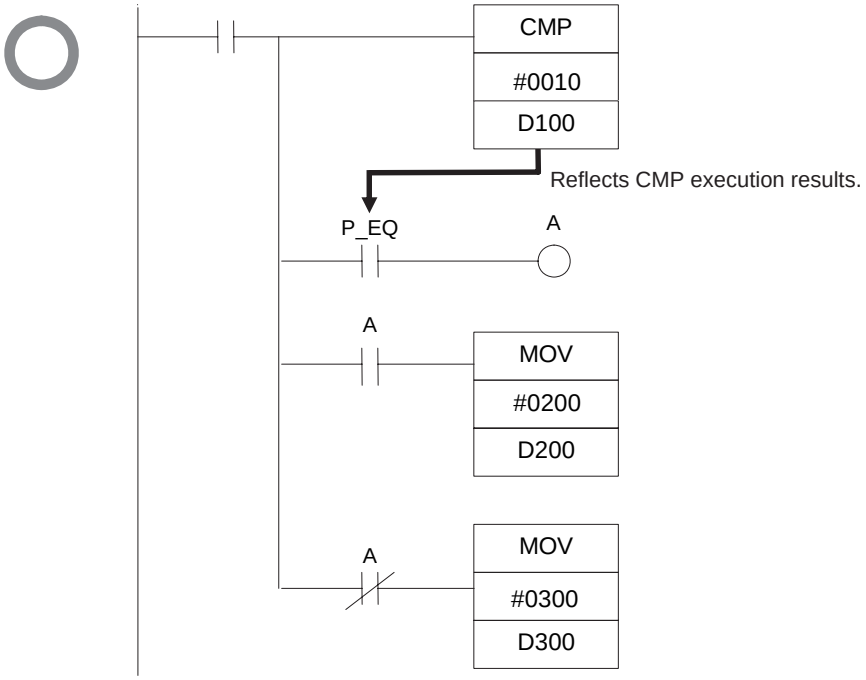
Make sure each of the results is picked up once by an OUTPUT instruction to ensure that execution results for instruction B will be not be picked up.



Example: The following example will move #0200 to D200 if D100 contains #0010 and move #0300 to D300 if D100 does not contain #0010.



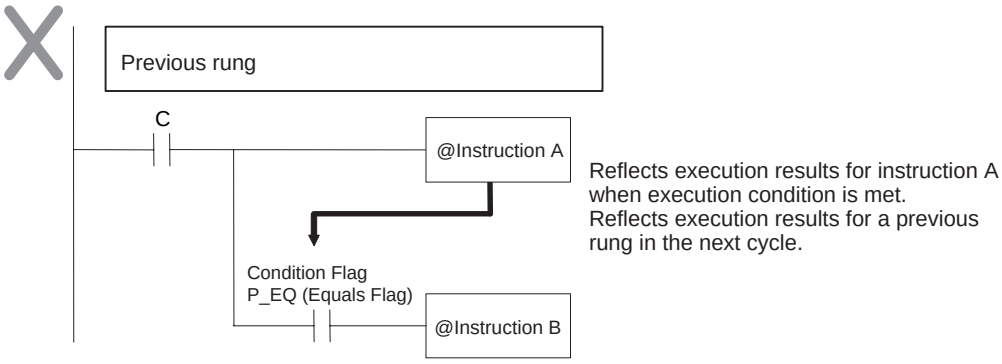
The Equals Flag will turn ON if D100 in the rung above contains #0010. #0200 will be moved to D200 for instruction (1), but then the Equals Flag will be turned OFF because the #0200 source data is not 0000 Hex. The MOV instruction at (2) will then be executed and #0300 will be moved to D300. A rung will therefore have to be inserted as shown below to prevent execution results for the first MOV instruction from being picked up.



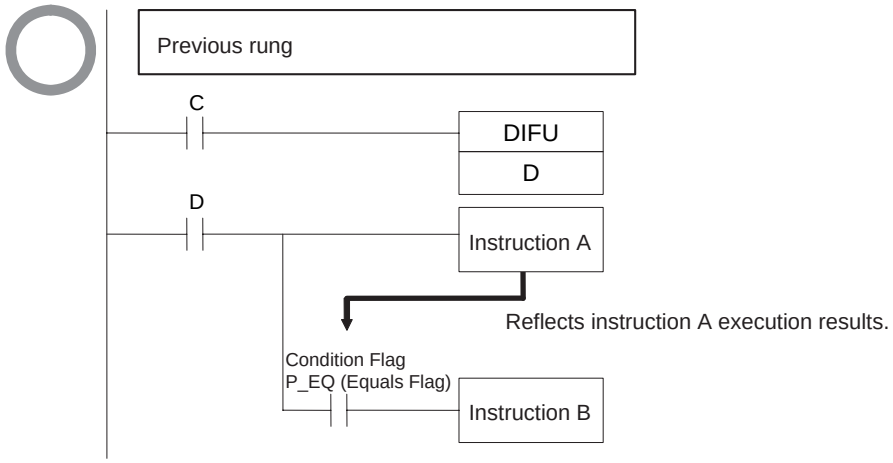
Using Execution Results from Differentiated Instructions

With differentiated instructions, execution results for instructions are reflected in Condition Flags only when execution condition is met, and results for a previous rung (rather than execution results for the differentiated instruction) will be reflected in Condition Flags in the next cycle. You must therefore be aware of what Condition Flags will do in the next cycle if execution results for differentiated instructions to be used.

In the following for example, instructions A and B will execute only if execution condition C is met, but the following problem will occur when instruction B picks up execution results from instruction A. If execution condition C remains ON in the next cycle after instruction A was executed, then instruction B will unexpectedly execute (by the execution condition) when the Condition Flag goes from OFF to ON because of results reflected from a previous rung.



In this case then, instructions A and B are not differentiated instructions, the DIFU (of DIFD) instruction is used instead as shown below and instructions A and B are both upwardly (or downwardly) differentiated and executed for one cycle only.



Additional Information

The CONDITION FLAG SAVE and CONDITION FLAG LOAD (CCS(282) and CCL(283)) instructions can be used to save and load the Condition Flag status. These can be used to access the status of the Condition Flags at other locations in a task or in a different task.

Main Conditions Turning ON Condition Flags

● Error Flag (P_ER)

The Error Flag will turn ON under special conditions, such as when operand data for an instruction is incorrect. The instruction will not be executed when the Error Flag turns ON.

When the Error Flag is ON, the status of other Condition Flags, such as the <, >, OF, and UF Flags, will not change and status of the = and N Flags will vary from instruction to instruction.

Refer to the descriptions of individual instructions in the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474) for the conditions that will cause the Error Flag to turn ON. Caution is required because some instructions will turn OFF the Error Flag regardless of conditions.

Additional Information

The PLC Setup Settings for when an instruction error occurs determines whether operation will stop when the Error Flag turns ON. In the default setting, operation will continue when the Error Flag turns ON. If Stop Operation is specified when the Error Flag turns ON and operation stops (treated as a program error), the program address at the point where operation stopped will be stored at in A298 to A299. At the same time, A295.08 will turn ON.

● Equals Flag (P_EQ)

The Equals Flag is a temporary flag for all instructions except when comparison results are equal (=). It is set automatically by the system, and it will change. The Equals Flag can be turned OFF (ON) by an instruction after a previous instruction has turned it ON (OFF). The Equals Flag will turn ON, for example, when MOV or another move instruction moves 0000 Hex as source data and will be OFF at all other times. Even if an instruction turns the Equals Flag ON, the move instruction will execute immediately and the Equals Flag will turn ON or OFF depending on whether the source data for the move instruction is 0000 Hex or not.

Carry Flag (P_CY)

The CY Flag is used in shift instructions, addition and subtraction instructions with carry input, addition and subtraction instruction borrows and carries, as well as with Special I/O Unit instructions, PID instructions, and FPD instructions. Note the following precautions.

- The CY Flag is used in shift instructions, addition and subtraction instructions with carry input, addition and subtraction instruction borrows and carries, as well as with Special I/O Unit instructions, PID instructions, and FPD instructions. Note the following precautions.
- The CY Flag can be turned ON (OFF) by the execution results for a certain instruction and be turned OFF (ON) by another instruction. Be sure the proper results are reflected in the Carry Flag when using it.

Less Than and Greater Than Flags (P_LT, P_GT)

The > and < Flags are used in comparison instruction, as well as in the LMT, BAND, ZONE, PID and other instructions.

The > or < Flag can be turned OFF (ON) by another instruction even if it is turned ON (OFF) by execution results for a certain instruction.

Negative Flag (P_N)

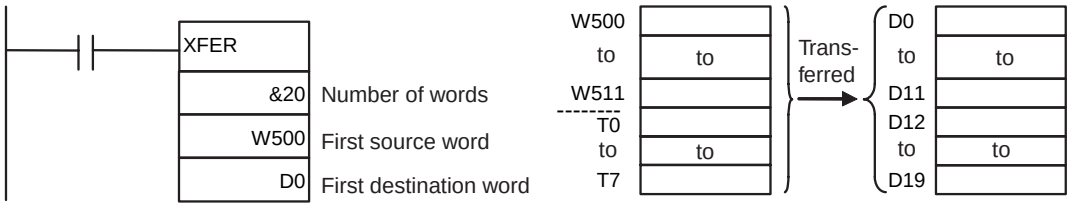
The N Flag is turned OFF when the leftmost bit of the instruction execution results word is “1” for certain instructions and it is turned OFF unconditionally for other instruction.

Specifying Operands for Multiple Words

An instruction will be executed as written even if an operand requiring multiple words is specified so that all of the words for the operand are not in the same area. In this case, words will be taken in order of the PLC memory addresses. The Error Flag will not turn ON.

As an example, consider the results of executing a block transfer with XFER(070) if 20 words are specified for transfer beginning with W500. Here, the Work Area, which ends at W511, will be exceeded, but the instruction will be executed without turning ON the Error Flag. In the PLC memory addresses, the present values for timers are held in memory after the Work Area, and thus for the following instruction, W500 to W511 will be transferred to D0 to D11 and the present values for T0 to T7 will be transferred to D12 to D19.

Note Refer to the appendix Memory Map of PLC Memory Addresses for specific PLC memory addresses.



5-10-2 Special Program Sections

CJ-series programs have special program sections that will control instruction conditions. The following special program sections are available

Program section	Instructions	Instruction condition	Status
Subroutine	SBS, SBN, and RET instructions GSBS, GSBN, and GRET instructions	Subroutine program is executed.	The subroutine program section between SBN and RET instructions is executed.
IL - ILC section	IL and ILC instructions	Section is interlocked	The output bits are turned OFF and timers are reset. Other instructions will not be executed and previous status will be maintained.
Step Ladder section	STEP instructions		
FOR-NEXT loop	FOR instructions and NEXT instructions	Break in progress.	Looping
JMP0 - JME0 section	JMP0 instructions and JME0 instructions		Jump
Block program section	BPRG instructions and BEND instructions	Block program is executing.	The block program listed in mnemonics between the BPRG and BEND instructions is executed.

Instruction Combinations

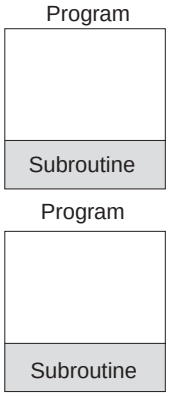
The following table shows which of the special instructions can be used inside other program sections.

	Subroutine	IL - ILC section	Step ladder section	FOR - NEXT loop	JMP0 - JME0 section	Block program section
Subroutine	Not possible.	Not possible.	Not possible.	Not possible.	Not possible.	Not possible.
IL - ILC	OK	Not possible.	Not possible.	OK	OK	Not possible.
Step ladder section	Not possible.	OK	Not possible.	Not possible.	OK	Not possible.
FOR - NEXT loop	OK	OK	Not possible.	OK	OK	Not possible.
JMP0 - JME0	OK	OK	Not possible.	Not possible.	Not possible.	Not possible.
Block program section	OK	OK	OK	Not possible.	OK	Not possible.

Note Instructions that specify program areas cannot be used for programs in other tasks. Refer to *5-2-4 Designing Tasks* for details.

Subroutines

Place all the subroutines together just before the END(001) instruction in all programs but after programming other than subroutines. (Therefore, a subroutine cannot be placed in a step ladder, block program, FOR - NEXT, or JMP0 - JME0 section.) If a program other than a subroutine program is placed after a subroutine program (SBN to RET), that program will not be executed.



Instructions Not Available in Subroutines

The following instructions cannot be placed in a subroutine.

Function	Mnemonic	Instruction
Process Step Control	STEP(008)	Define step ladder section
	SNXT(009)	Step through the step ladder

Note A subroutine can include a block program section. If, however, the block program is in WAIT status when execution returns from the subroutine to the main program, the block program section will remain in WAIT status the next time it is called.

Instructions Not Available in Step Ladder Program Sections

The following instructions cannot be used in step ladder program sections.

Function	Mnemonic	Instruction
Sequence Control	FOR(512), NEXT(513), and BREAK(514)	FOR, NEXT, and BREAK LOOP
	END(001)	END
	IL(002) and ILC(003)	INTERLOCK and INTERLOCK CLEAR
	JMP(004) and JME(005)	JUMP and JUMP END
	CJP(510) and CJPN(511)	CONDITIONAL JUMP and CONDITIONAL JUMP NOT
	JMP0(515) and JME0(516)	MULTIPLE JUMP and MULTIPLE JUMP END
Subroutines	SBN(092), RET(093), GSBN (751) and GRET(752)	SUBROUTINE ENTRY and SUBROUTINE RETURN
Block Programs	IF(802) (NOT), ELSE(803), and IEND(804)	CONDITIONAL BLOCK BRANCHING (NOT), CONDITIONAL BLOCK BRANCHING ELSE, and CONDITIONAL BLOCK BRANCHING END
	BPRG(096) and BEND(801)	BLOCK PROGRAM BEGIN/END
	EXIT(806) (NOT)	CONDITIONAL BLOCK EXIT (NOT)
	LOOP(809) and LEND(810) (NOT)	LOOP BLOCK and LOOP BLOCK END (NOT)
	WAIT(805) (NOT)	ONE CYCLE WAIT (NOT)
	TIMW(813)	TIMER WAIT
	TMHW(815)	HIGH-SPEED TIMER WAIT
	CNTW(814)	COUNTER WAIT
	BPPS(811) and BPRS(812)	BLOCK PROGRAM PAUSE and RESTART

Note 1 A step ladder program section can be used in an interlock section (between IL and ILC). The step ladder section will be completely reset when the interlock is ON.

2 A step ladder program section can be used between MULTIPLE JUMP (JMP0) and MULTIPLE JUMP END (JME0).

Instructions Not Supported in Block Program Sections

The following instructions cannot be placed in block program sections

Classification by Function	Mnemonic	Instruction
Sequence Control	FOR(512), NEXT(513), and BREAK(514)	FOR, NEXT, and BREAK LOOP
	IL(002) and ILC(003)	INTERLOCK and INTERLOCK CLEAR
	JMP0(515) and JME0(516)	MULTIPLE JUMP and MULTIPLE JUMP END
	END(001)	END
Sequence Input	UP(521)	CONDITION ON
	DOWN(522)	CONDITION OFF
Sequence Output	DIFU	DIFFERENTIATE UP
	DIFD	DIFFERENTIATE DOWN
	KEEP	KEEP
	OUT	OUTPUT
	OUT NOT	OUTPUT NOT
Timer/Counter	TIM	HUNDRED-MS TIMER
	TIMH	TEN-MS TIMER
	TMHH(540)	ONE-MS TIMER
	TIMU	TENTH-MS TIMER
	TMUH	HUNDREDTH-MS TIMER
	TTIM(087)	ACCUMULATIVE TIMER
	TIML(542)	LONG TIMER
	MTIM(543)	MULTI-OUTPUT TIMER
	CNT	COUNTER
	CNTR	REVERSIBLE COUNTER
Subroutines	SBN(092) and RET(093)	SUBROUTINE ENTRY and SUBROUTINE RETURN
Data Shift	SFT	SHIFT
Ladder Step Control	STEP(008) and SNXT(009)	STEP DEFINE and STEP START
Data Control	PID	PID CONTROL
Block Program	BPRG(096)	BLOCK PROGRAM BEGIN
Damage Diagnosis	FPD(269)	FAILURE POINT DETECTION

Note 1 Block programs can be used in a step ladder program section.

2 A block program can be used in an interlock section (between IL and ILC). The block program section will not be executed when the interlock is ON.

3 A block program section can be used between MULTIPLE JUMP (JMP0) and MULTIPLE JUMP END (JME0).

4 A JUMP instruction (JMP) and CONDITIONAL JUMP instruction (CJP/CJPN) can be used in a block program section. JUMP (JMP) and JUMP END (JME) instructions, as well as CONDITIONAL JUMP (CJP/CJPN) and JUMP END (JME) instructions cannot be used in the block program section unless they are used in pairs. The program will not execute properly unless these instructions are paired.

6

I/O Memory Areas

This section describes the I/O memory areas in the CPU Unit. I/O memory is one type of memory built into the CPU Unit.

6-1 I/O Memory Areas 6-2
6-1-1 I/O Memory Area Overview 6-2
6-1-2 I/O Memory Area Structure 6-4
6-1-3 Holding I/O Memory Values 6-6
6-2 I/O Area 6-8
6-2-1 Input Bits 6-8
6-2-2 Output Bits 6-10
6-3 Data Link Area 6-13
6-4 Synchronous Data Refresh Area 6-14
6-5 CPU Bus Unit Area 6-15
6-6 Special I/O Unit Area 6-16
6-7 Serial PLC Link Area 6-17
6-8 DeviceNet Area 6-18
6-9 Work Area 6-19
6-10 Holding Area 6-20
6-11 Auxiliary Area 6-22
6-12 Temporary Relay Area 6-23
6-13 Data Memory Area 6-24
6-14 Extended Data Memory Area 6-27
6-15 Timer Areas 6-31
6-16 Counter Areas 6-33
6-17 Task Flags 6-34
6-18 Index Registers 6-35
6-19 Data Registers 6-40
6-20 Condition Flags 6-42
6-21 Clock Pulses 6-44

6-1 I/O Memory Areas

6-1-1 I/O Memory Area Overview

I/O memory areas can be accessed using instruction operands. The following table lists the areas in I/O Memory.

Area name		Description	Reference
CIO Area (Core I/O Area)		Words in the CIO Area are used for data exchanges such as I/O refreshing with various Units. Words that are not allocated to Units may be used as work words and work bits in the program. It is not necessary to input the “CIO” prefix when specifying an address in the CIO Area.	---
	I/O Area	Words in the I/O Area are allocated to external I/O terminals on Basic I/O Units.	6-8
	Data Link Area	Words in the Data Link Area are used for data links with other PLC Units on a network through a Controller Link Unit.	6-13
	Synchronous Data Refresh Area	Words in the Synchronous Data Refresh Area are allocated as synchronous refresh data when synchronous unit operation is used.	6-14
	CPU Bus Unit Area	Words in the CPU Bus Unit Area are allocated to CPU Bus Units to transfer status information.	6-15
	Special I/O Unit Area	Words in the Special I/O Unit Area are allocated to Special I/O Units to transfer status information.	6-16
	Serial PLC Link Area	This area is used for Serial PLC Links. Data is exchanged between CPU Units using serial ports without communications programming. This area is supported only by CJ2M CPU Units.	6-17
	DeviceNet Area	Words in the DeviceNet Area are allocated to slaves for DeviceNet remote I/O communications. Allocations in this area are fixed and cannot be changed.	6-18
	Internal I/O Area *1	These words can be used only in the program. They cannot be used for I/O exchange with external I/O terminals.	---
Work Area (W) *1		Words in the Work Area can be used only in the program.	6-19
Holding Area (H)		Words in the Holding Area can be used only in the program. These words retain their content when the PLC is turned ON or the operating mode is switched between PROGRAM mode and RUN or MONITOR mode.	6-20
Auxiliary Area (A)		The Auxiliary Area contains flags and control bits used to monitor and control PLC operation.	6-22
Temporary Relay Area (TR) (TR Area)		The TR Area contains bits that record the ON/OFF status of program branches. The TR bits are used with mnemonics only.	6-23
Data Memory Area (D) (DM Area)		The DM Area is a multi-purpose data area. With CJ2 PLCs, the DM Area can be read and written in either word units or bit units. Words in the DM Area retain their contents when the PLC is turned ON or the operating mode is switched between PROGRAM mode and RUN or MONITOR mode. Some words in the DM Area are used to store initial settings for Special I/O Units and CPU Bus Units.	6-24
Extended Data Memory Area (E) (EM Area)		The EM Area is a multi-purpose data area. With CJ2 PLCs, the EM Area can be read and written in either word units or bit units. Words in the EM Area retain their content when the PLC is turned ON or the operating mode is switched between PROGRAM mode and RUN or MONITOR mode. The EM Area is divided into banks.	6-27
Timer Areas	Timer Completion Flag Area (T)	A Completion Flag is turned ON when the set time of the timer elapses.	6-31
	Timer PV Area (T)	The PV of a timer increases or decreases as the timer operates.	

Area name		Description	Reference
Counter Areas	Counter Completion Flag Area (C)	A Completion Flag is turned ON when the set value is reached or counted down to zero.	6-33
	Counter PV Area (C)	The PVs of a counter is incremented or decremented as the counter operates.	
Task Flag Area (TK)		A Task Flag is ON when the corresponding cyclic task is in READY status.	6-34
Index Registers (IR)		Index registers are used to store PLC memory addresses (absolute memory addresses in RAM) to indirectly address I/O memory words.	6-35
Data Registers (DR)		Data registers are used to offset the PLC memory addresses in Index Registers when addressing words indirectly.	6-40
Condition Flags		Condition Flags are special flags, such as the Error Flag and Carry Flag, that show the results of executing instructions.	6-42
Clock Pulses		The clock pulses are special flags that turn ON and OFF at regular intervals.	6-44

*1 There are two areas that provide work bits: The Internal I/O Area in the CIO Area and the Work Area. Use word bits in the Work Area first.

6-1-2 I/O Memory Area Structure

Area		Size	Range	External I/O allocation	Bit access	Word access	Access		Change from Programming Device	Status at startup or mode change	Forcing bit status	Reference
							Read	Write				
CIO Area	I/O Area	2,560 bits (160 words)	CIO 0 to CIO 159*1	Basic I/O Units	OK	OK	OK	OK	OK	Cleared*2	OK	6-8
	Data Link Area	3,200 bits (200 words)	CIO 1000 to CIO 1199	Data links or PLC links (conditional)	OK	OK	OK	OK	OK		OK	6-13
	Synchronous Data Refresh Area	1,536 bits (96 words)	CIO 1200 to CIO 1295	Synchronous Units*3	OK	OK	OK	OK	OK		OK	6-14
	CPU Bus Unit Area	6,400 bits (400 words)	CIO 1500 to CIO 1899	CPU Bus Units (conditional)	OK	OK	OK	OK	OK		OK	6-15
	Special I/O Unit Area	15,360 bits (960 words)	CIO 2000 to CIO 2959	Special I/O Units (conditional)	OK	OK	OK	OK	OK		OK	6-16
	Serial PLC Link Area	1,440 bits (90 words)	CIO 3100 to CIO 3189	Linked PLC	OK	OK	OK	OK	OK		OK	6-17
	DeviceNet Area	9,600 bits (600 words)	CIO 3200 to CIO 3799	DeviceNet Master (fixed allocations) (conditional)	OK	OK	OK	OK	OK		OK	6-18
	Internal I/O Area	3,200 bits (200 words) 37,504 bits (2,344 words)	CIO 1300 to CIO 1499 CIO 3800 to CIO 6143	---	OK	OK	OK	OK	OK		OK	---
Work Area		8,192 bits (512 words)	W000 to W511	---	OK	OK	OK	OK	OK	Cleared *2	OK	6-19
Holding Area*4		8,192 bits (512 words)	H000 to H511	---	OK	OK	OK	OK	OK	Maintained	OK	6-20
Auxiliary Area		48,128 bits (3,008 words)	A000 to A447	---	OK	OK	OK	No	No	Depends on the address	No	6-22
			A448 to A959	---	OK	OK	OK	OK	OK			
			A960 to A1471*5	---	OK	OK	OK	OK	OK			
			A10000 to A11535*5	---	OK	OK	OK	No	No			

*1 The I/O Area can be expanded to include CIO 0160 to CIO 0999 by changing the first words allocated to specified Units. Settings for the first words can be made using the CX-Programmer to set the first words in the I/O tables. The setting range for the first words is CIO 0 to CIO 900.

*2 If the I/O Memory Hold Flag (A500.12) is ON, the memory values will be maintained when the operating mode is changed. If, in addition, the PLC Setup is set to hold the status of the I/O Memory Hold Flag at startup (IOM Hold Bit parameter), the memory values will be maintained when the power supply is turned ON.

*3 This area is supported only by CJ2H CPU Units. “Synchronous Units” are CPU Bus Units and Special I/O Units that support synchronous unit operation.

*4 H512 to H1535 can be set for use only as function block memory or SFC memory.

*5 A960 to A1471 and A10000 to A11535 were added to expand the Auxiliary Area in CJ2 CPU Units. These words cannot be accessed by CPU Bus Units, Special I/O Units, PTs, and Support Software that do not specifically support the CJ2 CPU Units.
Only the following CPU Bus Units and Special I/O Units specifically support the CJ2 CPU Units.

- EtherNet/IP Unit: CJ1W-EIP21
- Position Control Units: CJ1W-NC214, CJ1W-NC234, CJ1W-NC281, CJ1W-NC414, CJ1W-NC434, CJ1W-NC481, and CJ1W-NC881
- Analog Input Unit: CJ1W-AD042
- Analog Output Unit: CJ1W-DA042V
- Serial Communications Units: CJ1W-SCU22, CJ1W-SCU32, and CJ1W-SCU42

Area	Size	Range	External I/O allocation	Bit access	Word access	Access		Change from Programming Device	Status at startup or mode change	Forcing bit status	Reference
						Read	Write				
TR Area	16 bits	TR0 to TR15	---	OK	---	OK	OK	No	Cleared	No	6-23
DM Area	32,768 words	D00000 to D32767	---	OK*7	OK	OK	OK	OK	Maintained	No	6-24
EM Area	32,768 words per bank, 25 banks max. (0 to 18 hex)	E00_0 to E18_32767*7	---	OK*7	OK	OK	OK	OK	Maintained	Can be enabled with a setting.*8	6-27
Timer Completion Flags	4,096 bits	T0 to T4095	---	OK	---	OK	OK	OK	Cleared	OK	6-31
Counter Completion Flags	4,096 bits	C0 to C4095	---	OK	---	OK	OK	OK	Maintained	OK	6-33
Timer PVs	4,096 words	T0 to T4095	---	---	OK	OK	OK	OK	Cleared	No*9	6-31
Counter PVs	4,096 words	C0 to C4095	---	---	OK	OK	OK	OK	Maintained	No*10	6-33
Task Flag Area	128 bits	TK000 to TK127	---	OK	---	OK	No	No	Cleared	No	6-34
Index Registers*6	16 registers	IR0 to IR15	---	OK	OK	Indirect addressing only	Specific instructions only	No	Cleared	No	6-35
Data Registers*6	16 registers	DR0 to DR15	---	No	OK	OK	OK	No	Cleared	No	6-40
Condition Flags	Example: Always ON Flag	System symbols in the global symbols table of the CX-Programmer (e.g., P_On)	---	OK	---	OK	No	No	Cleared	No	6-42
Pulse bits	Example: 1 s Clock Pulse	System symbols in the global symbols table of the CX-Programmer (e.g., P_1s)	---	OK	---	OK	No	No	Cleared	No	6-44

*6 Index registers and data registers can be used either individually by task or they can be shared by all the tasks.

*7 Banks D to 18 hex of the EM Area were added to expand the EM Area in CJ2 CPU Units. Also, the ability to address bits in the DM Area and EM Area was also added as a new feature to the CJ2 CPU Units.
Banks D to 18 hex of the EM Area cannot be accessed and bit addresses in the DM Area and EM Area cannot be used by CPU Bus Units, Special I/O Units, PTs, and Support Software that do not specifically support the CJ2 CPU Units.
Only the following CPU Bus Units and Special I/O Units specifically support the CJ2 CPU Units.

- EtherNet/IP Unit: CJ1W-EIP21
- Position Control Units: CJ1W-NC214, CJ1W-NC234, CJ1W-/NC281, CJ1W-NC414, CJ1W-NC434, CJ1W-NC481, and CJ1W-NC881
- Analog Input Unit: CJ1W-AD042
- Analog Output Unit: CJ1W-DA042V
- Serial Communications Units: CJ1W-SCU22, CJ1W-SCU32, and CJ1W-SCU42

*8 Bits in the specified bank and all banks after it can be force-set/reset. (This is called the EM Area force-set/reset function.) With CJ2H CPU Units, bits in following EM Area banks that are set for automatic address allocation can be force-set/reset.
CJ2H-CPU64/65(-EIP): E03_0 to E03_32767
CJ2H-CPU65(-EIP): E06_0 to E09_32767
CJ2H-CPU67(-EIP): E07_0 to E0E_32767
CJ2H-CPU68(-EIP): E11_0 to E18_32767

*9 Timer PVs can be refreshed indirectly by force-setting/resetting Timer Completion Flags.

*10 Counter PVs can be refreshed indirectly by force-setting/resetting Counter Completion Flags.

6-1-3 Holding I/O Memory Values

Clearing I/O Memory for CPU Unit Operating Status Changes

I/O memory values (except for some I/O memory areas) are cleared when the operating status of the CPU Unit changes.

● Changes for Which I/O Memory Is Cleared

- Changing the operating mode between PROGRAM mode and RUN or MONITOR Mode
- Stopping operation due to a fatal error (except for stopping operation due to executing a FALS(007) instruction
- Resetting the power supply (turning the power OFF and then back ON)

● Settings to Hold I/O Memory

You can make settings to hold I/O memory even if the operating status of the CPU Unit changes.

- Holding I/O Memory for Operating Mode Changes and Fatal Errors:
  Turn ON the IOM Hold Bit in the Auxiliary Area (A500.12).
- Holding I/O Memory When Resetting Power
  Turn ON the IOM Hold Bit in the Auxiliary Area (A500.12) and select the *IOM Hold Bit* Check Box in the Startup Hold Area on the Startup Tab Page in the PLC Setup.



Precautions for Safe Use

Making Settings to Hold the I/O Area

Output bits in I/O memory will not be cleared (i.e., will not turn OFF) when the CPU Unit's operating mode changes from RUN or MONITOR mode to PROGRAM mode. The status before changing to PROGRAM mode will be held. When the mode is then changed from PROGRAM mode to RUN or MONITOR mode, the previous I/O memory values will be output.

When operation stops due to a fatal error (including execution of the FALS(007) instruction), the I/O memory values in the CPU Unit will be held, but the outputs from the Output Units will all turn OFF.

I/O Memory Hold Status According to I/O Memory Area

The following table gives the hold status for each I/O memory area.

Area		CPU Unit operating mode changes	Fatal errors		Power supply turned ON
			FALS(007) executed	Other fatal errors	
CIO Area (Core I/O Area)	I/O Area	Possible to hold*2 (Depends on settings.)	Held	Possible to hold*2 (Depends on settings.)	Possible to hold*3 (Depends on settings.)
	Data Link Area				
	Synchronous Data Refresh Area*1				
	CPU Bus Unit Area				
	Special I/O Unit Area				
	Serial PLC Link Area*4				
	DeviceNet Area				
	Internal I/O Area				
Work Area (W)		Possible to hold*2 (Depends on settings.)	Held	Possible to hold*2 (Depends on settings.)	Possible to hold*3 (Depends on settings.)
Holding Area (H)		Held			
Auxiliary Area (A)		Held or cleared depending on the address.			
Data Memory Area (D)		Held			
Extended Data Memory Area (E)		Held			
Timer Completion Flags (T)		Possible to hold*2 (Depends on settings.)	Held	Possible to hold*2 (Depends on settings.)	Possible to hold*3 (Depends on settings.)
Timer PVs (T)		Possible to hold*2 (Depends on settings.)	Held	Possible to hold*2 (Depends on settings.)	Possible to hold*3 (Depends on settings.)
Counter Completion Flags (C)		Held			
Counter PVs (C)		Held			
Task Flag Area (TK)		Cleared	Held	Cleared	Cleared
Index Registers (IR)		Cleared	Held	Cleared	Cleared
Data Registers (DR)		Cleared	Held	Cleared	Cleared

*1 This area is supported only by CJ2H CPU Units.

*2 Turn ON the IOM Hold Bit (A500.12) in the Auxiliary Area to hold these areas.

*3 Turn ON the IOM Hold Bit in the Auxiliary Area (A500.12) and select the IOM Hold Bit Check Box in the Startup Hold Area on the Startup Tab Page in the PLC Setup.

6-2 I/O Area

I/O Area addresses range from CIO 0 to CIO 159 for words and CIO 0.00 to CIO 159.15 for bits.

Words in the I/O Area are allocated to I/O terminals on Basic I/O Units. Words are allocated to Basic I/O Units based on the slot position (left to right) and number of words required. The words are allocated consecutively and empty slots are skipped. Words in the I/O Area that aren't allocated to Basic I/O Units can be used only in the program.

Bits in the I/O Area can be force-set and force-reset.

Note The area can be expanded to CIO 0000 to CIO 0999 by changing the first word allocated to a specific Unit. The maximum number of bits that can be allocated for external I/O will still be 2,560 (160 words) even if the I/O Area is expanded.

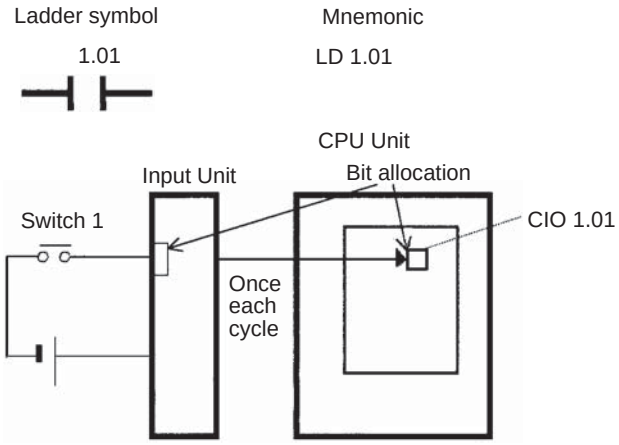
6-2-1 Input Bits

A bit in the I/O Area is called an input bit when it is allocated to an Input Unit. Input bits reflect the ON/OFF status of devices such as pushbutton switches, limit switches, and photoelectric switches.

There are three ways for the status of input points to be refreshed in the PLC: normal I/O refreshing, immediate refreshing, and IORF(097) refreshing.

Normal I/O Refreshing

The status of input points on external devices is read once each cycle after program execution. In the following example, CIO 1.01 is allocated to switch 1, an external switch connected to the input terminal of an Input Unit. The ON/OFF status of switch 1 is stored in CIO 1.01 once each cycle.



Immediate Refreshing

When the immediate refreshing variation of an instruction is specified by inputting an exclamation point just before the instruction and the instruction's operand is an input bit or word, refreshing is performed as described below. This immediate refreshing is performed in addition to the normal I/O refreshing performed once each cycle.

Bit Operands

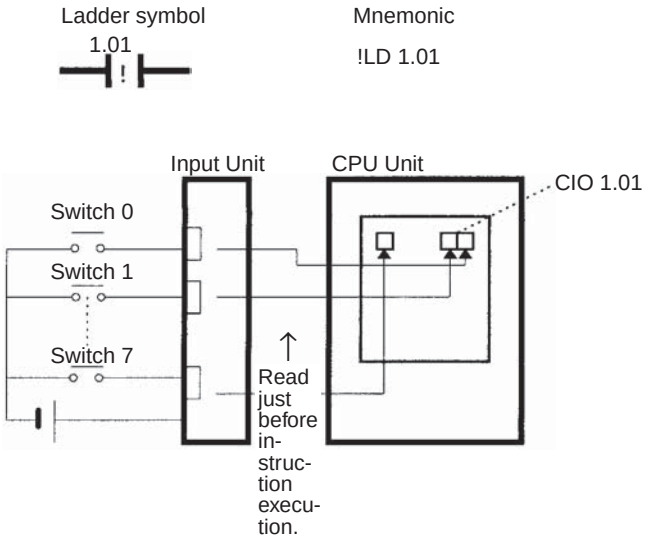
Just before the instruction is executed, the ON/OFF status of the 16 I/O points allocated to the word containing the specified bit will be read to the PLC.

Word Operands

Just before the instruction is executed, the ON/OFF status of the 16 I/O points allocated to the specified word will be read to the PLC.

Example

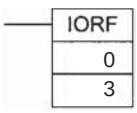
In the following example, CIO 1.01 is allocated to switch 1, an external switch connected to the input terminal of an Input Unit. The ON/OFF status of switch 1 is read and reflected in CIO 1.01 just before !LD 1.01 is executed.



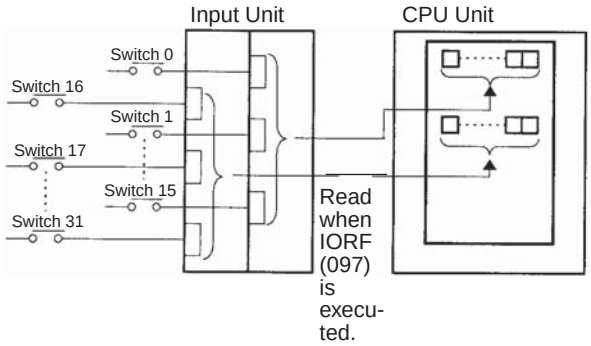
IORF(097) Refreshing

When the IORF(097) (I/O REFRESH) instruction is executed, the input bits in the specified range of words are refreshed. This I/O refreshing is performed in addition to the normal I/O refreshing performed once each cycle.

The following IORF(097) instruction refreshes the status of all I/O points in I/O Area words CIO 0 to CIO 3. The status of input points is read from the Input Units and the status of output bits is written to the Output Units.



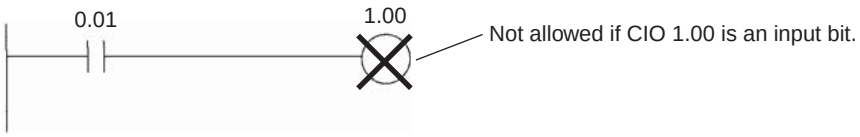
In the following example, the status of input points allocated to CIO 0 and CIO 1 are read from the Input Unit. (CIO 2 and CIO 3 are allocated to Output Units.)



Restrictions on Input Bits

There is no limit on the number of times that input bits can be used as normally open and normally closed conditions in the program. The addresses can be programmed in any order.

An input bit cannot be used as an operand in an OUTPUT instruction.



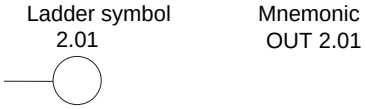
6-2-2 Output Bits

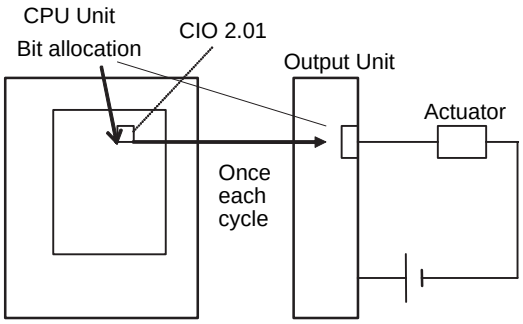
A bit in the I/O Area is called an output bit when it is allocated to an Output Unit. The ON/OFF status of output bits are output to devices such as actuators. There are three ways for the status of output bits to be refreshed to an Output Unit: normal I/O refreshing, immediate refreshing, and IORF(097) refreshing.

Normal I/O Refreshing

The status of output bits are output to external devices once each cycle after program execution.

In the following example, CIO 2.01 is allocated to an actuator, an external device connected to an output terminal of an Output Unit. The ON/OFF status of CIO 2.01 is output to that actuator once each cycle.





Immediate Refreshing

When the immediate refreshing variation of an instruction is specified by inputting an exclamation point just before the instruction, and the instruction's operand is an output bit or word, refreshing is performed as described below. This immediate refreshing is performed in addition to the normal I/O refreshing performed once each cycle.

Bit Operands

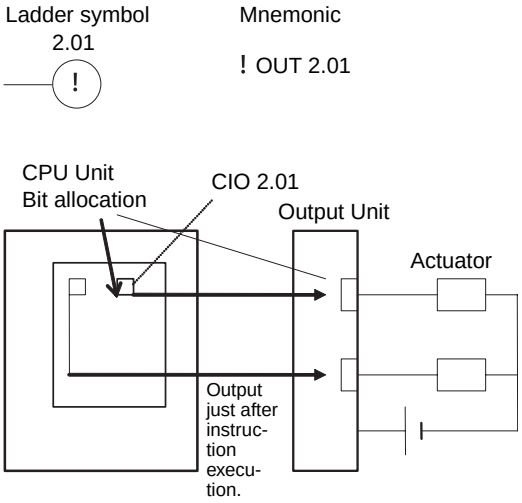
Just after the instruction is executed, the ON/OFF status of the 16 I/O points allocated to the word containing the specified bit will be output to the output devices.

Word Operands

Just after the instruction is executed, the ON/OFF status of the 16 I/O points allocated to the specified word will be output to the output devices.

Example

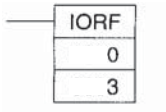
In the following example, CIO 2.01 is allocated to an actuator, an external device connected to the output terminal of an Output Unit. The ON/OFF status of CIO 2.01 is output to the actuator just after !OUT 2.01 is executed.



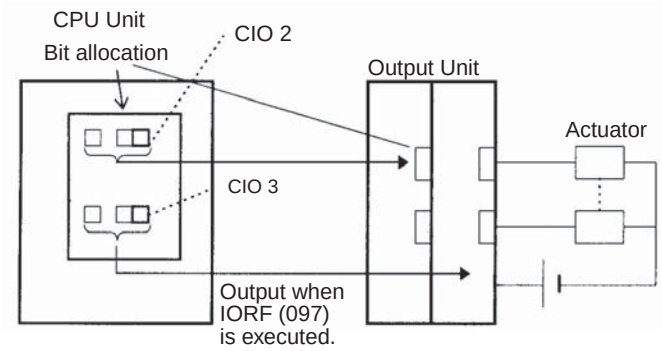
IORF(097) Refreshing

When IORF(097) (I/O REFRESH) is executed, the ON/OFF status of output bits in the specified range of words is output to their external devices. This I/O refreshing is performed in addition to the normal I/O refreshing performed once each cycle.

The following IORF(097) instruction refreshes the status of all I/O points in I/O Area words CIO 0 to CIO 3. The status of input points is read from the Input Units and the status of output bits is written to the Output Units.



In this example, the status of input points allocated to CIO 2 and CIO 3 are output to the Output Unit. (CIO 0 and CIO 1 are allocated to Input Units.)

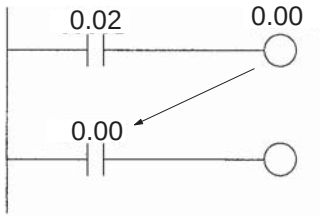


Precautions for Correct Use

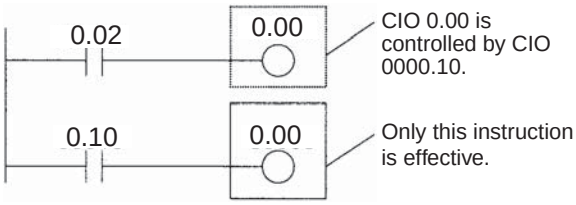
All outputs on Basic I/O Units and Special I/O Units can be turned OFF by turning ON the Output OFF Bit (A500.15). The status of the output bits will not be affected even though the actual outputs are turned OFF.

Restrictions on Output Bits

Output bits can be programmed in any order. Output bits can be used as operands in input instructions. There is no limit on the number of times that output bit can be used as a normally open and normally closed conditions in the program.



An output bit can be used in only one instruction that controls its status. If the status of an output bit is controlled by two or more instructions, only the last instruction will be effective.



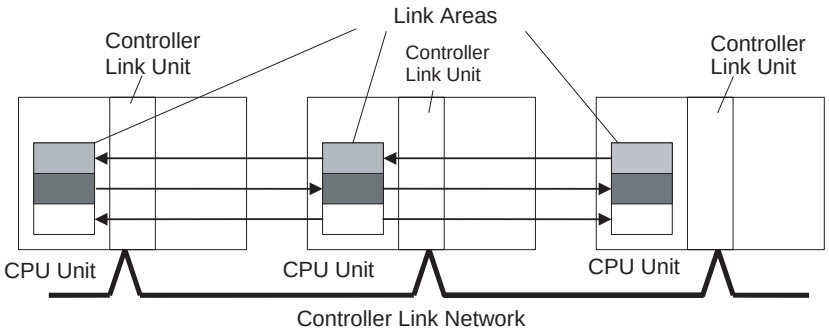
6-3 Data Link Area

Data Link Area addresses range from CIO 1000 to CIO 1199 for words and CIO 1000.00 to CIO 1199.15 for bits. Words in the Data Link Area are used for data links when *LR* is set as the data link area for Controller Link Networks.

A data link automatically (independently of the program) shares data with Data Link Areas in other CPU Units in the network through Controller Link Units mounted to the PLCs.

Data links can be generated automatically (using the same number of words for each node) or manually. When a user defines the data links manually, any number of words can be assigned to each node and nodes can be made receive-only or send-only. Refer to the *Controller Link Units Operation Manual* (Cat. No. W309) for details.

Words in the Data Link Area can be used in the program when *LR* is not set as the data link area for Controller Link Networks are not used.



Bits in the Data Link Area can be force-set and force-reset.

6-4 Synchronous Data Refresh Area

Synchronous Data Refresh Area addresses range from CIO 1200 to CIO 1295 for words and CIO 1200.00 to CIO 1295.15 for bits.

The Synchronous Data Refresh Area is used to exchange data between the CPU Unit and Synchronous Units when synchronous unit operation is used for CJ2H CPU Units. This area is supported only by CJ2H CPU Units. Refer to *10-8-4 Synchronous Data Refresh* for details.

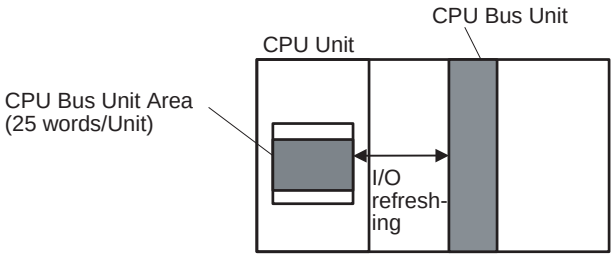
6-5 CPU Bus Unit Area

The CPU Bus Unit Area addresses range from CIO 1500 to CIO 1899 for words and CIO 1500.00 to CIO 1899.15 for bits.

Words in the CPU Bus Unit Area are allocated to CPU Bus Units to transfer data, such as the operating status of the Unit. Each Unit is allocated 25 words based on the Unit's unit number setting.

Data is exchanged with CPU Bus Units at the following times.

- During the I/O refresh period
- When DLNK(226)) is executed



Each CPU Bus Unit is allocated 25 words based on its unit number, as shown in the following table.

Unit number	Allocated words
0	CIO 1500 to CIO 1524
1	CIO 1525 to CIO 1549
2	CIO 1550 to CIO 1574
3	CIO 1575 to CIO 1599
4	CIO 1600 to CIO 1624
5	CIO 1625 to CIO 1649
6	CIO 1650 to CIO 1674
7	CIO 1675 to CIO 1699
8	CIO 1700 to CIO 1724
9	CIO 1725 to CIO 1749
A	CIO 1750 to CIO 1774
B	CIO 1775 to CIO 1799
C	CIO 1800 to CIO 1824
D	CIO 1825 to CIO 1849
E	CIO 1850 to CIO 1874
F	CIO 1875 to CIO 1899

The function of the 25 words depends on the CPU Bus Unit being used. For details, refer to the Unit's operation manual.

Words in the CPU Bus Unit Area that are not allocated to CPU Bus Units can be used only in the program.

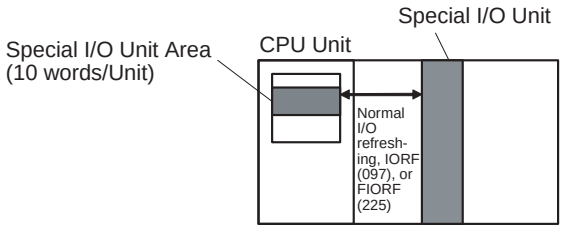
6-6 Special I/O Unit Area

The Special I/O Unit Area addresses range from CIO 2000 to CIO 2959 for words and CIO 2000.00 to CIO 2959.15 for bits.

Words in the Special I/O Unit Area are allocated to Special I/O Units for data, such as the operating status of each Unit. Each Unit is allocated 10 words based on its Unit's unit number setting. Up to 96 Units can be used with unit numbers 0 to 95.

Words in the Special I/O Unit Area are refreshed at the following times:

- Normal I/O refreshing
- When IORF (097) is executed
- When FIORF (225) is executed



Each Special I/O Unit is allocated 25 words based on its unit number, as shown in the following table.

Unit number	Allocated words
0	CIO 2000 to CIO 2009
1	CIO 2010 to CIO 2019
2	CIO 2020 to CIO 2029
3	CIO 2030 to CIO 2039
4	CIO 2040 to CIO 2049
5	CIO 2050 to CIO 2059
6	CIO 2060 to CIO 2069
7	CIO 2070 to CIO 2079
8	CIO 2080 to CIO 2089
9	CIO 2090 to CIO 2099
10 (A)	CIO 2100 to CIO 2109
11 (B)	CIO 2110 to CIO 2119
12 (C)	CIO 2120 to CIO 2129
13 (D)	CIO 2130 to CIO 2139
14 (E)	CIO 2140 to CIO 2149
15 (F)	CIO 2150 to CIO 2159
16	CIO 2160 to CIO 2169
17	CIO 2170 to CIO 2179
⋮	⋮
95	CIO 2950 to CIO 2959

The function of the 10 words allocated to a Unit depends on the Special I/O Unit being used. For details, refer to the Unit's operation manual.

Words in the Special I/O Unit Area that are not allocated to Special I/O Units can be used only in the program.

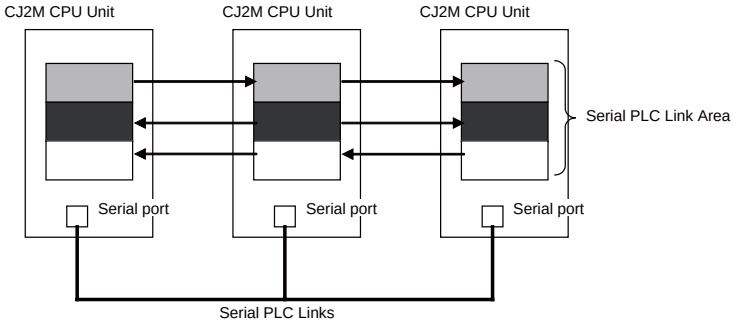
6-7 Serial PLC Link Area

Serial PLC Link Area addresses range from CIO 3100 to CIO 3189 for words and CIO 3100.00 to CIO 3189.15 for bits.

The Serial PLC Link Area is used for Serial PLC Links. They can be used for data links to other PLCs. For Serial PLC Links, data is exchanged between CPU Units using serial ports without communications programming.

The words in the Serial PLC Link Area are allocated automatically according to the settings made in the PLC Setup of the polling PLC.

- Serial PLC Link mode
- Serial PLC Link number of send words
- Serial PLC Link maximum unit number



Words in the Serial PLC Link Area that are not allocated to Serial PLC Links can be used only in the program as work bits. Bits in the Serial PLC Link Area can be force-set and force-reset.

The contents of this area will be cleared in the following cases:

- The operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa.
- When the PLC power is reset
- When the area is cleared from the CX-Programmer
- When operation stops due to a fatal error (except that the area is not cleared when stopping operation due to execution of a FALS(007) instruction)

● Setting the IOM Hold Bit (A500.12) to ON

If the IOM Hold Bit (A500.12) is ON, the contents of this area will be retained when a fatal error occurs or the operating mode is changed from PROGRAM mode to RUN or MONITOR mode or vice-versa. However, the contents will be cleared when power is cycled.

● Setting the IOM Hold Bit (A500.12) to ON and Protecting the IOM Hold Bit itself in the PLC Setup

The contents of this area will be maintained even when the power is cycled.

6-8 DeviceNet Area

The DeviceNet Area address range from CIO 3200 to CIO 3799 for words and CIO 3200.00 to CIO 3799.15 for bits.

Words in the DeviceNet Area are allocated to Slaves for DeviceNet remote I/O communications. Data is exchanged regularly with slaves in the network (independent of the program) through the DeviceNet Unit.

Words are allocated to slaves using fixed allocations according to fixed allocation settings 1, 2, and 3. One of these fixed areas is selected.

Area	Output Area (master to slaves)	Input Area (slaves to master)
Fixed Allocation Area 1	CIO 3200 to CIO 3263	CIO 3300 to CIO 3363
Fixed Allocation Area 2	CIO 3400 to CIO 3463	CIO 3500 to CIO 3563
Fixed Allocation Area 3	CIO 3600 to CIO 3663	CIO 3700 to CIO 3763

The following words are allocated to the DeviceNet Unit when the remote I/O slave function is used with fixed allocations.

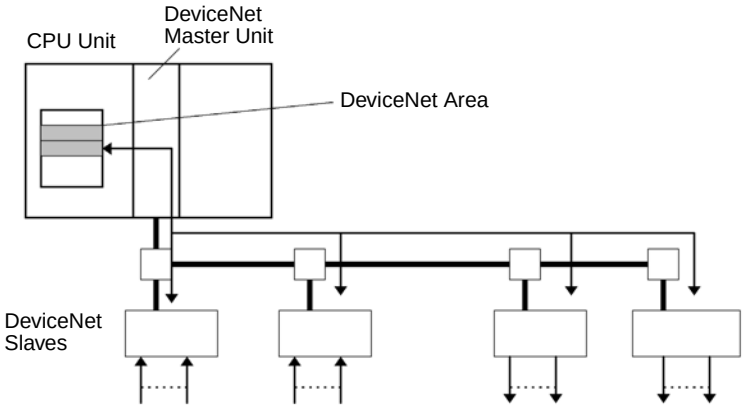
Area	Output Area (master to slaves)	Input Area (slaves to master)
Fixed Allocation Area 1	CIO 3370	CIO 3270
Fixed Allocation Area 2	CIO 3570	CIO 3470
Fixed Allocation Area 3	CIO 3770	CIO 3670

Additional Information

There are two ways to allocate I/O in DeviceNet networks: Fixed allocations according to node addresses and user-set allocations.

- With fixed allocations, words are automatically allocated to the slaves in the specified fixed allocation area (1 to 3) according to the node addresses.
- With user-set allocations, the user can allocate words to Slaves from the following words.
  CIO 0 to CIO 6143
  W0 to W511
  H0 to H511
  D0 to D32767
  E00_0 to E0C_32767, Banks: 0 to 18 hex
  (You cannot specify an EM bank that is higher than C.)

For details on word allocations, refer to the *DeviceNet Operation Manual* (Cat. No. W267).



With fixed allocation, words are assigned according to node numbers. (If a Slave requires two or more words, it will occupy as many node numbers as required to obtain enough words.)

6-9 Work Area

The Work Area contains 512 words with addresses ranging from W0 to W511 for words and W0.00 to W511.15 for bits. Words in this area cannot be used for I/O with external I/O terminals. They can be used internally in the program.

There are also unused words in the Internal I/O Area in the CIO Area (CIO 1300 to CIO 1499 and CIO 3800 to CIO 6143) that can also be used in the program. The unused words in the CIO Area, however, may be allocated to new functions in future versions of the CPU Units. Always use any available words in the Work Area first, before using words in the Internal I/O Area.

6-10 Holding Area

The Holding Area addresses range from H000 to H511 for words and H000.00 to H511.15 for bits. These words can be used only in the program. Bits in this area will maintain the status when power is turned OFF or when the operating mode is changed from PROGRAM mode to RUN or MONITOR mode or vice-versa.

Holding Area bits can be used in any order in the program and can be used as normally open or normally closed conditions as often as necessary.

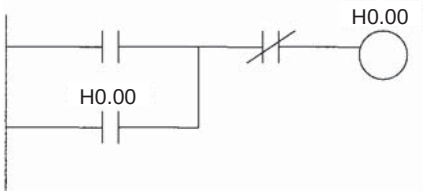
● Holding Area Initialization

Data in the Holding Area is not cleared when the PLC's power supply is cycled or the CPU Unit operating mode is changed from PROGRAM mode to RUN or MONITOR mode or vice-versa.

A Holding Area bit will be cleared if it is programmed between IL(002) and ILC(003) and the execution condition for IL(002) is OFF. To keep a bit ON even when the execution condition for IL(002) is OFF, turn ON the bit with the SET instruction just before IL(002).

● Self-maintaining Bits

When a self-maintaining bit is programmed with a Holding Area bit, the self-maintaining bit won't be cleared even when the power is reset.



If a Holding Area bit is not used for the self-maintaining bit, the bit will be turned OFF and the self-maintaining bit will be cleared when the power is reset.

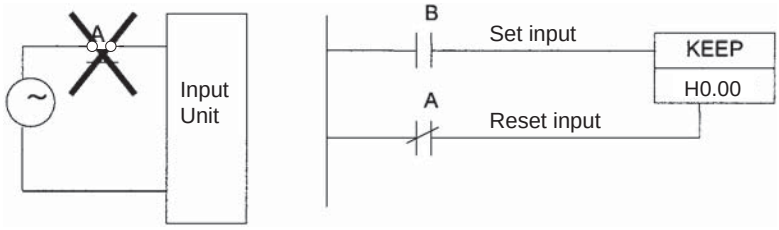
If a Holding Area bit is used but not programmed as a self-maintaining bit as in the following diagram, the bit will be turned OFF by execution condition A when the power is reset.



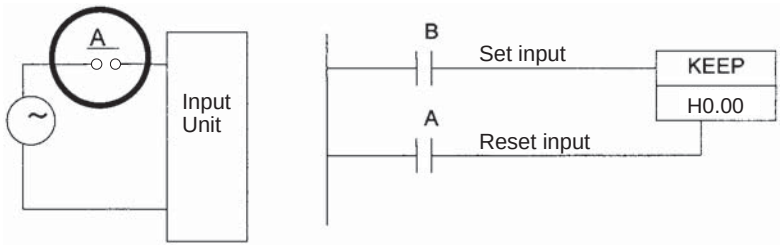
The Holding Area words from H512 to H1535 can be set for use with function blocks or SFC programs.These words cannot be specified as instruction operands in the user program.

● Precautions

When a Holding Area bit is used in a KEEP(011) instruction, never use a normally closed condition for the reset input if the input device uses an AC power supply. When the power supply goes OFF or is temporarily interrupted, the input will go OFF before the PLC's internal power supply and the Holding Area bit will be reset.



Instead, use a configuration like the one shown below.



There are no restrictions in the order of using bit address or in the number of N.C. or N.O. conditions that can be programmed.

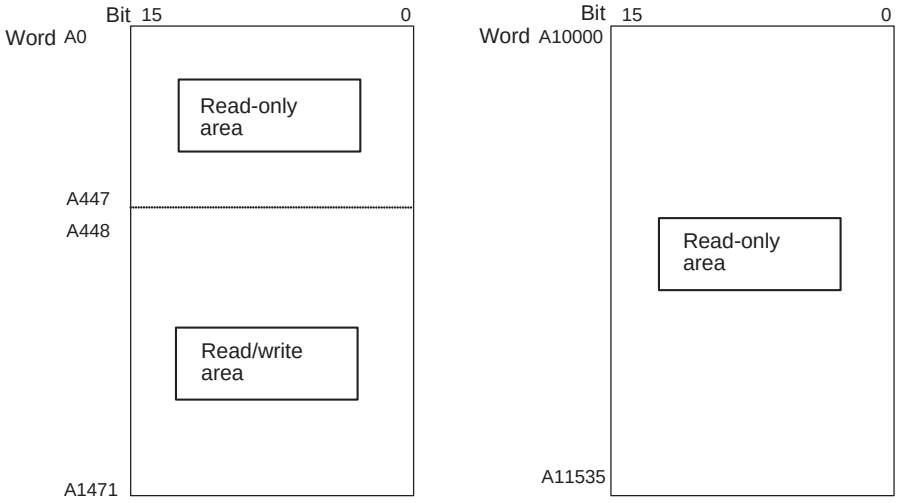
6-11 Auxiliary Area

The Auxiliary Area addresses range from A0 to A1471 and A1000 to A11535 for words, and A0.00 to A1471.15 and A1000.00 to A11535.15 for bits. These words are preassigned as flags and control bits to monitor and control operation.

Some words or bits are set automatically by the system and others are set and manipulated by the user. The Auxiliary Area includes error flags set by self-diagnosis, initial settings, control bits, and status data. (For information the functions of Auxiliary Area bits and words, refer to the *A-3 Auxiliary Area*.)

A0 through A447 and A10000 through A11535 are read-only, but A448 through A1471 can be read or written from the program or the CX-Programmer.
A960 to A1471 and A10000 to A11535 were added to expand the Auxiliary Area in CJ2 CPU Units. These words cannot be accessed by CPU Bus Units, Special I/O Units, PTs, and Support Software that do not specifically support the CJ2 CPU Units.
Only the following CPU Bus Units and Special I/O Units specifically support the CJ2 CPU Units.

- EtherNet/IP Unit: CJ1W-EIP21
- Position Control Units: CJ1W-NC214, CJ1W-NC234, CJ1W-NC281, CJ1W-NC414, CJ1W-NC434, CJ1W-NC481, and CJ1W-NC881
- Analog Input Unit: CJ1W-AD042
- Analog Output Unit: CJ1W-DA042V
- Serial Communications Units: CJ1W-SCU22, CJ1W-SCU32, and CJ1W-SCU42



The following operations can be performed from the CX-Programmer to write data in the Auxiliary Area: Changing present values when monitoring programming addresses (set values dialog box), or transferring data to the PLC after editing the PLC data tables. Also the online set/reset operation cannot be used for bits in the Auxiliary Area. Refer to the *CX-Programmer Operation Manual* (Cat. No. W414).

Precautions for Correct Use

There is a possibility that a function will be assigned to any undefined Auxiliary Area word or bit in a future upgrade of the CPU Unit. Do not use undefined words or bits in the Auxiliary Area as work words or bits in the user program.

6-12 Temporary Relay Area

The TR Area contains 16 bits with addresses ranging from TR0 to TR15. TR bits are useful when there are several output branches and interlocks cannot be used.

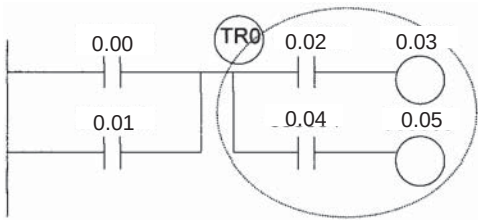
The TR bits can be used as many times as required and in any order required as long as the same TR bit is not used twice in the same instruction block.

TR bits can be used only with the OUT and LD instructions. OUT instructions (OUT TR0 to OUT TR15) store the ON OFF status of a branch point and LD instructions recall the stored ON OFF status of the branch point.

TR bit status cannot be changed using the CX-Programmer.

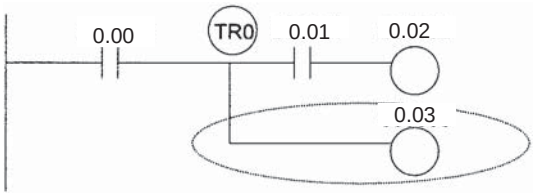
● Application Examples

In this example, a TR bit is used when two outputs have been directly connected to a branch point.



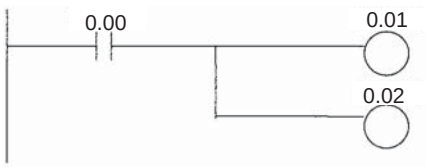
Instruction	Operand
LD	0.00
OR	0.01
OUT	TR 0
AND	0.02
OUT	0.03
LD	TR 0
AND	0.04
OUT	0.05

In this example, a TR bit is used when an output is connected to a branch point without a separate execution condition.

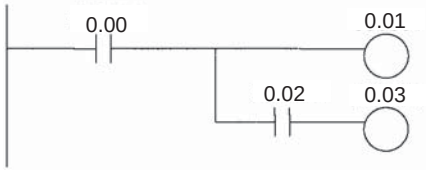


Instruction	Operand
LD	0.00
OUT	TR 0
AND	0.01
OUT	0.02
LD	TR 0
OUT	0.03

A TR bit is not required when there are no execution conditions after the branch point or there is an execution condition only in the last line of the instruction block.



Instruction	Operand
LD	0.00
OUT	0.01
OUT	0.02



Instruction	Operand
LD	0.00
OUT	0.01
AND	0.02
OUT	0.03

6-13 Data Memory Area

The DM Area addresses range from D0 to D32767 for words. This data area is used for general data storage and manipulation and is accessible by word or bit.

Data in the DM Area is retained when the PLC's power is cycled or the CPU Unit operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa.

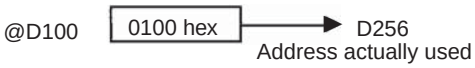
Bits in the DM Area cannot be force-set or force-reset.

Indirect Addressing

Words in the DM Area can be indirectly addressed in two ways: binary-mode and BCD-mode.

● Binary-mode Addressing (@D)

When a "@" character is input before a DM address, the content of that DM word is treated as binary and the instruction will operate on the DM word at that binary address. The entire DM Area (D0 to D32767) can be indirectly addressed with hexadecimal values 0 to 7FFF.



● BCD-mode Addressing (∗D)

When a "∗" character is input before a DM address, the content of that DM word is treated as BCD and the instruction will operate on the DM word at that BCD address. Only part of the DM Area (D0 to D09999) can be indirectly addressed with BCD values 0 to 9999.

Example: [MOV #1234 ∗D100]



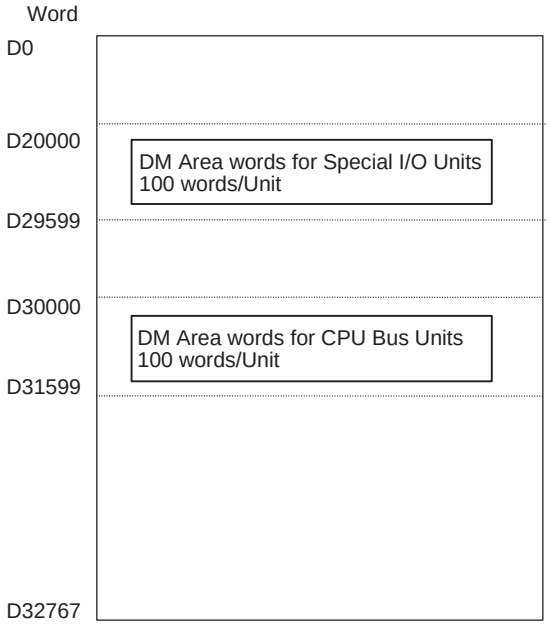
DM Area Allocations to Special I/O Units and CPU Bus Units

Parts of the DM Area are allocated to Special I/O Units and CPU Bus Units for functions, such as initial Unit settings. These words can be used for general data storage if the corresponding Unit is not used in the PLC.

The timing for data transfers is different for these Units, but may occur at any of the three following times.

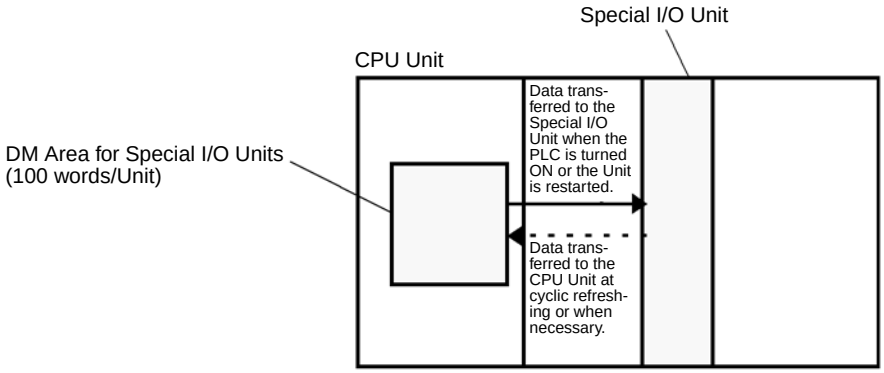
- Transferring data when the PLC's power is turned ON or the Unit is restarted
- Transferring data once each cycle
- Transferring data when required

Refer to the Unit's operation manual for details on data transfer timing.



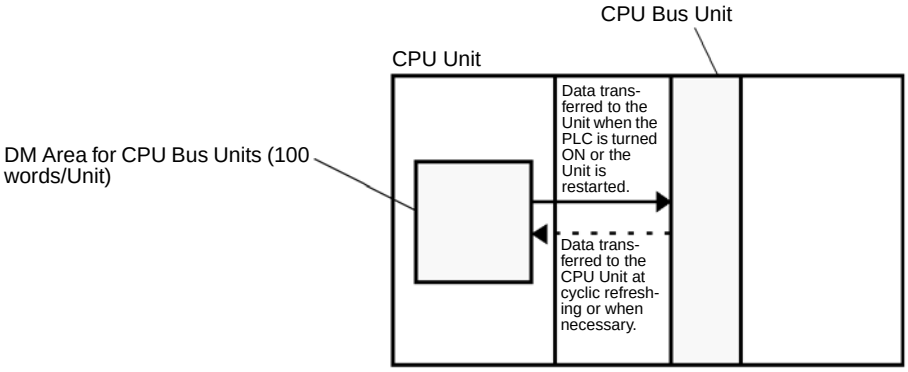
Special I/O Units (D20000 to D29599)

Each Special I/O Unit is allocated 100 words (based on unit numbers 0 to 95). Refer to the Unit's operation manual for details on the function of these words.



CPU Bus Units (D30000 to D31599)

Each CPU Bus Unit is allocated 100 words (based on unit numbers 0 to F). Refer to the Unit's operation manual for details on the function of these words. With some CPU Bus Units, such as Ethernet Units, initial settings must also be registered in the CPU Unit's Parameter Area; this data can be registered with a Programming Device.



6-14 Extended Data Memory Area

The EM Area addresses range from E0_0 to E18_32767 for words. The EM Area is divided into 24 banks (0 to 18 hex). The hexadecimal number after “E” in the address indicates the bank.

The EM Area is used for general data storage and manipulation and is accessible by word or bit.

Data in the EM Area is retained when the PLC's power is cycled or the CPU Unit operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa.

In addition to a general data area, settings can be made to use the EM Area for automatic address allocation, trace memory, and file memory.
Banks D to 18 hex of the EM Area (E0D_0 to E18_32767) were added to expand the EM Area in CJ2 CPU Units. These banks cannot be accessed by CPU Bus Units, Special I/O Units, PTs, and Support Software that do not specifically support the CJ2 CPU Units.
Only the following CPU Bus Units and Special I/O Units specifically support the CJ2 CPU Units.

- EtherNet/IP Unit: CJ1W-EIP21
- Position Control Units: CJ1W-NC214, CJ1W-NC234, CJ1W-NC281, CJ1W-NC414, CJ1W-NC434, CJ1W-NC481, and CJ1W-NC881
- Analog Input Unit: CJ1W-AD042
- Analog Output Unit: CJ1W-DA042V
- Serial Communications Units: CJ1W-SCU22, CJ1W-SCU32, and CJ1W-SCU42

Automatic Address Allocation

We recommend that you use the EM Area to automatically allocate addresses to symbols. With CJ2H CPU Units, you can force-set/reset bits in the following EM Area banks if you use automatic address allocation for them.

Model	Words in which bits can be force-set/reset when using automatic address allocation
CJ2H-CPU64(-EIP)	E03_0 to E03_32767
CJ2H-CPU65(-EIP)	
CJ2H-CPU66(-EIP)	E06_0 to E09_32767
CJ2H-CPU67(-EIP)	E07_0 to E0E_32767
CJ2H-CPU68(-EIP)	E11_0 to E18_32767

Refer to *5-5-8 Automatic Address Allocation to Symbols* for the procedure to use automatic address allocation.

Additional Information

Bits in all EM Area banks can be force-set or force-reset for CPU Units with unit version 1.2 or later.

With the CJ2M CPU Units, force-setting/resetting bits in the EM Area is possible only for banks specified for the EM Area force-set/reset function without using automatic address allocation.

Models	Force-set/reset enabled banks
CJ2M-CPU□1	E0
CJ2M-CPU□2	
CJ2M-CPU□3	
CJ2M-CPU□4	E0 to E3
CJ2M-CPU□5	

EM Area Force-set/reset Function

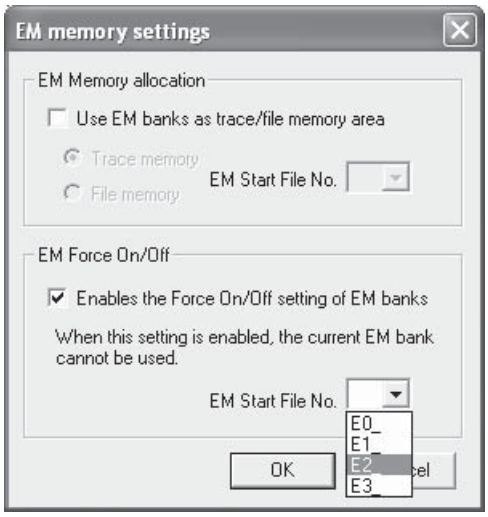
A parameter can be set from the CX-Programmer to enable force-setting/resetting bits in specified EM Area bank and all following EM Area banks. (This function is disabled in the default settings.)
This enables force-setting/resetting bits in the EM Area even if automatic address allocation areas are not used. This function can be set for all EM Area banks.

● Setting Procedure

1. Select ***PLC - Memory Allocate - EM Memory Settings*** from the CX-Programmer.*

The EM Memory Settings Dialog Box will be displayed.

2. Select the check box to enable the EM Area force-set/reset function and set the first EM Area bank.



3. Connect online to the CPU Unit and transfer the user program.

* CX-Programmer version 8.3 or higher is required to use the EM Area force-set/reset function.

Precautions for Correct Use

- When the EM Area force-set/reset function is enabled, the banks specified for the EM Area force-set/reset function cannot be used as the current EM Area bank.
- The EM Area force-set/reset function cannot be used for EM Area banks that are being used as trace memory or file memory.
- Force-setting and force-resetting are possible in MONITOR or PROGRAM mode. They cannot be executed in RUN mode.
  Refer to *10-7-1 Forced Set/Reset* for the procedure to use to force-set/reset bits.

File Memory

File memory can be used to store files used by the CPU Unit. Refer to *Section 7 File Operations* for details.

Trace Memory

Trace memory is used to data sampled for data tracing. Refer to *10-7-6 Tracing Data* for details.

Directly Specifying EM Addresses

There are two ways to specify an EM address: the bank and address can be specified at the same time or an address in the current bank can be specified. In general, we recommend specifying both the bank and address at the same time.

● Bank and Address Specification

With this method, the bank number is specified just before the address. For example, E2_10 specifies address E10 in bank 2.

● Current Bank Address Specification

With this method, just the address is specified. For example, E10 specifies address E10 in the current bank.

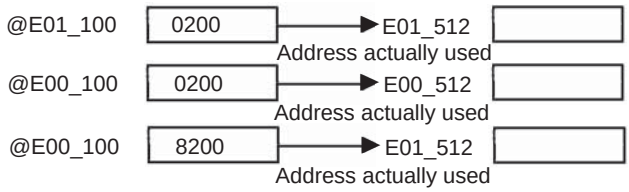
- The current bank will be reset to 0 when the operating mode is changed from PROGRAM mode to RUN/MONITOR mode, unless the IOM Hold Bit (A500.12) is ON.
- The current bank can be changed with EMBC(281) to access data in another bank.
- A301 contains the current EM bank number.
- The current bank is not changed as the program proceeds through cyclic tasks. For example, if the current EM bank is changed to bank 2 in task 1, the current EM bank will still be bank 2 in task 2. The current bank will return to its original value (in the source cyclic task) if it has been changed in an interrupt task.

Indirectly Specifying EM Addresses

Words in the EM Area can be indirectly addressed in two ways: binary-mode and BCD-mode.

● Binary-mode Addressing (@E)

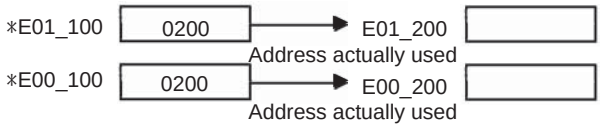
When a "@" character is input before an EM address, the content of that EM word is treated as a binary value indicating another EM Area address. The instruction will operate on the EM word at that binary address in the same bank or the next bank. All of the words in the same EM bank (E0 to E32767) can be indirectly addressed with hexadecimal values 0 to 7FFF and words in the next EM bank (E0 to E32767) can be addressed with hexadecimal values 8000 to FFFF.



● BCD-mode Addressing (*E)

When a "*" character is input before an EM address, the content of that EM word is treated as a BCD value indicating another EM address.

If the BCD value is between 0 and 9999, the final address will be in the same bank.



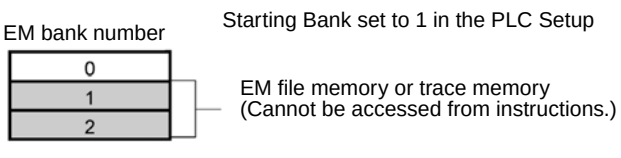
● Converting EM Area to File Memory or Trace Memory

You can specify a bank in the EM Area using the CX-Programmer to convert all banks from the specified bank to the end of EM Area to file memory or trace memory.

File Memory:
File memory can be used to store files, like storing files in a Memory Card. A maximum of 13 banks can be converted to file memory.

Trace Memory:
The results of a data trace can be saved in trace memory, increasing the quantity of results that can be stored for one trace.

Once EM Area banks have been converted to file memory or trace memory, those banks cannot be accessed from instructions in the user program. An Illegal Access Error will occur if you attempt to do so.



6-15 Timer Areas

Up to 4,096 timers with timer numbers T0 to T4095 can be used. There are two timer data areas: the Timer Completion Flag Area and the Timer Present Value (PV) Area.

- Timer Completion Flags (T)
  Timer numbers are used to access Completion Flags. A Completion Flag is turned ON when the set time of the timer elapses.
- Timer PVs (T)
  Timer numbers are also used to read and write the present values of timers (16-bit words). The PV increases or decreases as the timer operates.

When a timer number is used in an operand that requires bit data, the timer number accesses the Completion Flag. When a timer number is used in an operand that requires word data, the timer number accesses the PV.

The same timer numbers are used by all of the following instructions: HUNDRED-MS TIMER (TIM/TIMX(550)), TEN-MS TIMER (TIMH(015)/TIMHX(551)), ONE-MS TIMER (TMHH(540)/TMHHX(552)), TENTH-MS TIMER (TIMU(541)/TIMUX(556)), HUNDREDTH-MS TIMER (TMUH(544)/TMUHX(557)), ACCUMULATIVE TIMER (TTIM(087)/TTIMX(555)), TIMER WAIT (TIMW(813)/TIMWX(816)), and HIGH-SPEED TIMER WAIT (TMHW(815)/TMHWX(817)).



Precautions for Correct Use

It is not recommended to use the same timer number in two timer instructions because the timers will not operate correctly if they are timing simultaneously. (If two or more timer instructions use the same timer number, an error will be generated during the program check, but the timers will operate as long as the instructions are not executed in the same cycle.)

The following table shows when timer PVs and Completion Flags will be reset.

Instruction name*1	Effect on PV and Completion Flag			Operation in Jumps and Interlocks	
	Mode change (PROGRAM to RUN or MONITOR or vice-versa)*2	PLC startup*3	TRSET(549)	Jumps (JMP-JME) or tasks in WAIT status*4	Interlocks (IL-ILC)
HUNDRED-MS TIMER: TIM/TIMX(550)	PV → 0 Flag → OFF	PV → 0 Flag → OFF	PV → Maintained Flag → OFF	PVs refreshed in operating timers.	PV → SV (Reset to SV.) Flag → OFF
TEN-MS TIMER: TIMH(015)/TIMHX(551)					
ONE-MS TIMER: TMHH(540)/TMHHX(552)					
TENTH-MS TIMER: TIMU(541)/TIMUX(556)*1					
HUNDERDTH-MS TIMER: TMUH(544)/TMUHX(557)*1					
ACCUMULATIVE TIMER: TTIM(087)/TTIMX(555)				PV maintained.	PV maintained.
TIMER WAIT: TIMW(813)TIMWX(816)				PVs refreshed in operating timers.	---
HIGH-SPEED TIMER WAIT: TMHW(815)/TMHWX(817)					---

*1 The TIML(542), TIMLX(553), MTIM(543), and MTIMX(554) instructions do not use timer numbers, and they are reset under different conditions. (TIML(542) and TIMLX(553) are reset to their set values, and MTIM(543) and MTIMX(554) are reset to 0.) Refer to the descriptions of these instructions for details.

*2 If the IOM Hold Bit (A500.12) is ON, the PV and Completion Flag will be retained when a fatal error occurs or the operating mode is changed from PROGRAM mode to RUN or MONITOR mode or vice-versa. The PV and Completion Flag will be cleared when power is cycled. Refer to the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474) for details.

*3 If the IOM Hold Bit (A500.12) is ON and the IOM Hold Bit Check Box is selected in the Startup Hold Area on the Startup Tab Page in the PLC Setup, the PV and Completion Flag will be retained when the PLC's power is cycled.

*4 The present values of HUNDRED-MS TIMER (TIM/TIMX(550)), TEN-MS TIMER (TIMH(015)/TIMHX(551)), ONE-MS TIMER (TMHH(540)/TMHHX(552)), TIMER WAIT (TIMW(813)/TIMWX(816), and HIGH-SPEED TIMER WAIT (TMHW(815)/TMHWX(817)) timers programmed with timer numbers T0 to T2047 will be updated even when jumped between JMP and JME instructions or when in a task that is in WAIT status. The present value of timers programmed with timer numbers T2048 to T4095 will be held when jumped or when in a task that is in WAIT status.

- Timer Completion Flags can be force-set and force-reset.
- There are no restrictions in the order of using timer numbers or in the number of N.C. or N.O. conditions that can be programmed.
- The present data of all timers except for TENTH-MS TIMER (TIMU(541)/TIMUX(556)) and HUNDERDTH-MS TIMER (TMUH(544)/TMUHX(557)) can be read as word data.

6-16 Counter Areas

Up to 4,096 counters with counter numbers C0 to C4095 can be used. There are two counter data areas: the Counter Completion Flag Area and the Counter Present Value (PV) Area.

- Counter Completion Flags (C)
  Counter numbers are used to access Completion Flags. A Completion Flag is turned ON when the set value of the counter is reached.
- Counter PVs (C)
  Counter numbers are also used to read and write the present values of counters (16 bits). The PVs count up or down as the counter operates.

When a counter number is used in an operand that requires bit data, the counter number accesses the Completion Flag of the counter. When a counter number is used in an operand that requires word data, the counter number accesses the PV of the counter.

The same timer number are used by all of the following instructions CNT, CNTX(546), CNTR(012), CNTRX(548), CNTW(814), and CNTWX(818).



Precautions for Correct Use

It is not recommended to use the same counter number in two counter instructions because the counters will not operate correctly if they are counting simultaneously. If two or more counter instructions use the same counter number, an error will be generated during the program check, but the counters will operate as long as the instructions are not executed in the same cycle.

The following table shows when counter PVs and Completion Flags will be reset.

Instruction name	Effect on PV and Completion Flag					
	Reset	Mode change	PLC startup	Reset Input	CNR(545)/CNRX(547)	Interlocks (IL-ILC)
COUNTER: CNT/CNTX(546)	PV → 0000 Flag → OFF	Maintained	Maintained	Reset	Reset	Maintained
REVERSIBLE COUNTER: CNTR(012)/CNTRX(548)						
COUNTER WAIT: CNTW(814)/CNTWX(818)						

- Counter Completion Flags can be force-set and force-reset.
- Counter PVs cannot be force-set or force-reset, although the PVs can be refreshed indirectly by force-setting/resetting the Completion Flag.
- There are no restrictions in the order of using counter numbers or in the number of N.C. or N.O. conditions that can be programmed.
- Counter PVs can be read as word data and used in programming.

6-17 Task Flags

Task Flags range from TK0 to TK127 and correspond to cyclic tasks 0 to 127.

A Task Flag will be ON when the corresponding cyclic task is in READY or RUN status and OFF when the cyclic task is in INI or WAIT status.

Note These flags indicate the status of cyclic tasks only, they do not reflect the status of extra cyclic tasks or interrupt tasks.

The Task Flags will be cleared in the following cases, regardless of the status of the IOM Hold Bit (A500.12).

- The operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa.
- The PLC's power supply is cycled.

The Task Flags cannot be force-set or force-reset.

6-18 Index Registers

The sixteen Index Registers (IR0 to IR15) are used for indirect addressing. Each Index Register can hold a single PLC memory address, which is the absolute memory address of a word in I/O memory. These are different from the I/O memory area addresses in the CIO Area, DM Area, etc. They are the continuous RAM addresses.

Index Registers can be used either independently in each task or shared by all the tasks.

The user cannot directly input PLC memory addresses in the Index Registers. Use MOVR(560) to convert a regular data area address to its equivalent PLC memory address and write that value to the specified Index Register. (Use MOVRW(561) to set the PLC memory address of a timer/counter PV in an Index Register.)

Additional Information

Refer to *A-4 Memory Map of PLC Memory Addresses* for more details on PLC memory addresses.

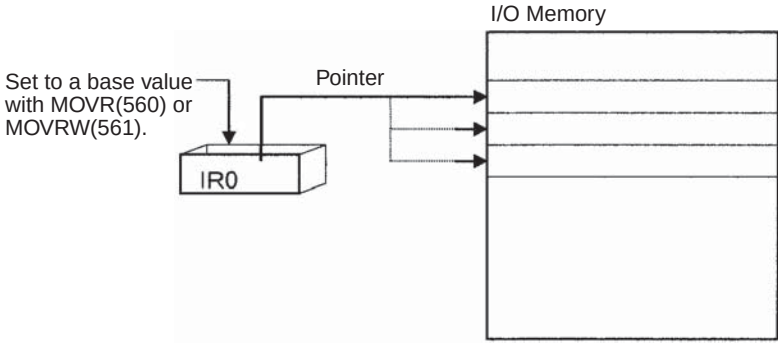
Indirect Addressing

When an Index Register is used as an operand with a "," prefix, the instruction will operate on the word indicated by the PLC memory address in the Index Register, not the Index Register itself. Basically, the Index Registers are I/O memory pointers.

- All addresses in I/O memory (except Index Registers, Data Registers, and Condition Flags) can be specified seamlessly with PLC memory addresses. It isn't necessary to specify the data area.
- In addition to basic indirect addressing, the PLC memory address in an Index Register can be offset with a constant or Data Register, auto-incremented, or auto-decremented. These functions can be used in loops to read or write data while incrementing or decrementing the address by one each time that the instruction is executed.

With the offset and increment/decrement variations, the Index Registers can be set to base values with MOVR(560) or MOVRW(561) and then modified as pointers with each instruction execution.

For pointer operation, either special instructions that can directly specify Index Registers (MOVR(560), MOVRW(561), increment instructions, decrement instructions, or math instructions) are used, or indirect offsets, auto-incrementing, or auto-decremented is used.



You can set the PLC to use index registers independently in each task or to share them between all tasks.

Index registers are cleared at the following times:

- When the operating mode is changed between PROGRAM or MONITOR mode and RUN mode
- When the power supply to the PLC is turned ON
- When a fatal error occurs (except for fatal errors created with FALS(007)

Precautions for Correct Use

It is possible to specify regions outside of I/O memory and generate an Illegal Access Error when indirectly addressing memory with Index Registers. Refer to *A-4 Memory Map of PLC Memory Addresses* for details on the limits of PLC memory addresses.

The following table shows the variations available when indirectly addressing I/O memory with Index Registers.

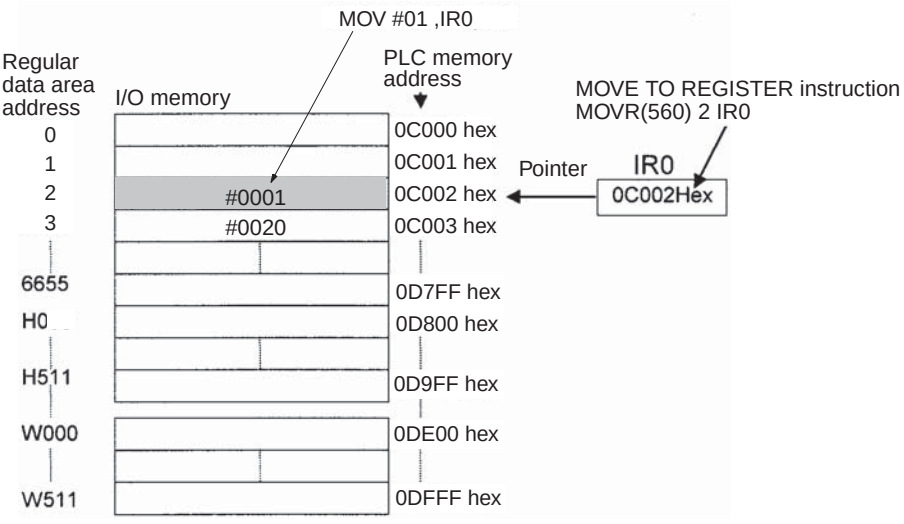
Variation	Function	Syntax	Example	
Indirect addressing	The content of IR□ is treated as the PLC memory address of a bit or word.	,IR□	LD , IR0	Loads the bit at the PLC memory address contained in IR0.
Indirect addressing with constant offset	The constant prefix is added to the content of IR□ and the result is treated as the PLC memory address of a bit or word. The constant may be any integer from –2,048 to 2,047.	Constant ,IR□ (Include a + or – in the constant.)	LD +5, IR0	Adds 5 to the contents of IR0 and loads the bit at that PLC memory address.
Indirect addressing with DR offset	The content of the Data Register is added to the content of IR□ and the result is treated as the PLC memory address of a bit or word.	DR□ , IR□	LD DR0 , IR0	Adds the contents of DR0 to the contents of IR0 and loads the bit at that PLC memory address.
Indirect addressing with auto-increment	After referencing the content of IR□ as the PLC memory address of a bit or word, the content is incremented by 1 or 2.	Increment by 1: , IR□+ Increment by 2: , IR□++	LD , IR0++	Loads the bit at the PLC memory address contained in IR0 and then increments the content of IR0 by 2.
Indirect addressing with auto-decrement	The content of IR□ is decremented by 1 or 2 and the result is treated as the PLC memory address of a bit or word.	Decrement by 1: ,– IR□ Decrement by 2: ,– – IR□	LD , – –IR0	Decrements the content of IR0 by 2 and then loads the bit at that PLC memory address.

Note IR□ represents an Index Register from IR0 to IR15.

Example

This example shows how to store the PLC memory address of a word (CIO 2) in an Index Register (IR0), use the Index Register in an instruction, and use the auto-increment variation.

MOVR(560) 2 IR0	Stores PLC memory address of CIO 2 in IR0.
MOV(021) #1 ,IR0	Writes #1 to PLC memory address contained in IR0, i.e., to CIO 2.
MOV(021) #20 +1, IR0	Reads IR0, adds 1, and writes #20 to that PLC memory address, i.e., to CIO 3.



Note The PLC memory addresses are listed in the diagram above, but it isn't necessary to know the PLC memory addresses when using Index Registers.

Some operands are treated as word data and others are treated as bit data, so the meaning of the data in an Index Register will differ depending on the operand in which it is used.

- Word Operand:
  MOVR(560) 0 IR2
  MOV(021) D0 , IR2

  When the operand is treated as a word, the contents of the Index Register are used "as is" as the PLC memory address of a word. In this example MOVR(560) sets the PLC memory address of CIO 2 in IR2 and the MOV(021) instruction copies the contents of D0 to CIO 2.

- Bit Operand:
  MOVR(560) 0.13 ,IR2
  SET +5 , IR2

  Index registers can also be used to specify bits, like in the SET instruction above. In this example, MOVR(560) sets the PLC memory bit address of CIO 0.13 in IR2. The SET instruction adds +5 from bit 13 to this PLC memory address, so it turns ON bit CIO 1.02.



Additional Information

When MOVR(560) is used to set a word address in an index register, the address is stored as follows:



When MOVR(560) is used to set a bit address in an index register, the address is stored as follows:



Direct Addressing

When an Index Register is used as an operand without a "," prefix, the instruction will operate on the contents of the Index Register itself (a two-word or "double" value). Index Registers can be directly addressed only in the instructions shown in the following table. Use these instructions to operate on the Index Registers as pointers.

The Index Registers cannot be directly addressed in any other instructions, although they can usually be used for indirect addressing.

Instruction group	Instruction name	Mnemonic
Data Movement Instructions	MOVE TO REGISTER	MOVR(560)
	MOVE TIMER/COUNTER PV TO REGISTER	MOVRW(561)
	DOUBLE MOVE	MOVL(498)
	DOUBLE DATA EXCHANGE	XCGL(562)
Table Data Processing Instructions	SET RECORD LOCATION	SETR(635)
	GET RECORD NUMBER	GETR(636)
Tracking Instructions	Unsigned One-word Record Search Instructions	RSRCH (360 to 364)
	UNSIGNED ONE-WORD RECORD SORT	RSORT(203)
Increment/Decrement Instructions	DOUBLE INCREMENT BINARY	++L(591)
	DOUBLE DECREMENT BINARY	– –L(593)
Comparison Instructions	DOUBLE EQUAL	=L(301)
	DOUBLE NOT EQUAL	< >L(306)
	DOUBLE LESS THAN	< L(311)
	DOUBLE LESS THAN OR EQUAL	< =L(316)
	DOUBLE GREATER THAN	> L(321)
	DOUBLE GREATER THAN OR EQUAL	> =L(326)
	DOUBLE COMPARE	CMPL(060)
Symbol Math Instructions	DOUBLE SIGNED BINARY ADD WITHOUT CARRY	+L(401)
	DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY	–L(411)

The SRCH(181), MAX(182), and MIN(183) instructions can output the PLC memory address of the word with the desired value (search value, maximum, or minimum) to IR0. In this case, IR0 can be used in later instructions to access the contents of that word.

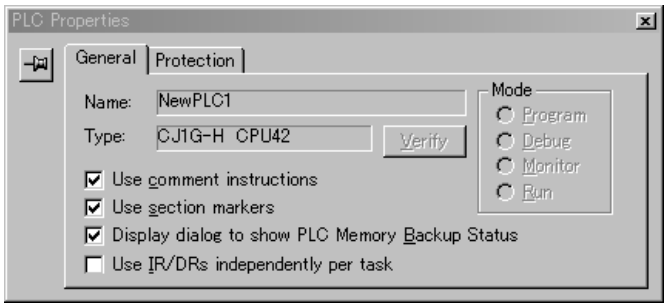
Precautions

- Do not use Index Registers until a PLC memory address has been set in the register. The pointer operation will be unreliable if the registers are used without setting their values.
- The values in Index Registers are unpredictable at the start of an interrupt task. When an Index Register will be used in an interrupt task, always set a PLC memory address in the Index Register with MOVR(560) or MOVRW(561) before using the register in that task.
- Index Registers are processed independently in each task, so they do not affect each other. For example, IR0 used in Task 1 and IR0 used in Task 2 are different. Consequently, each Index Register task has 16 Index Registers.
- It is possible to read the Index Register for only the last task executed within the cycle from the CX-Programmer. If using Index Registers with the same number to perform multiple tasks, it is only possible with the CX-Programmer to read the Index Register value for the last task performed within the cycle from the multiple tasks. Nor is it possible to write the Index Register value from the CX-Programmer.

Sharing Index Registers

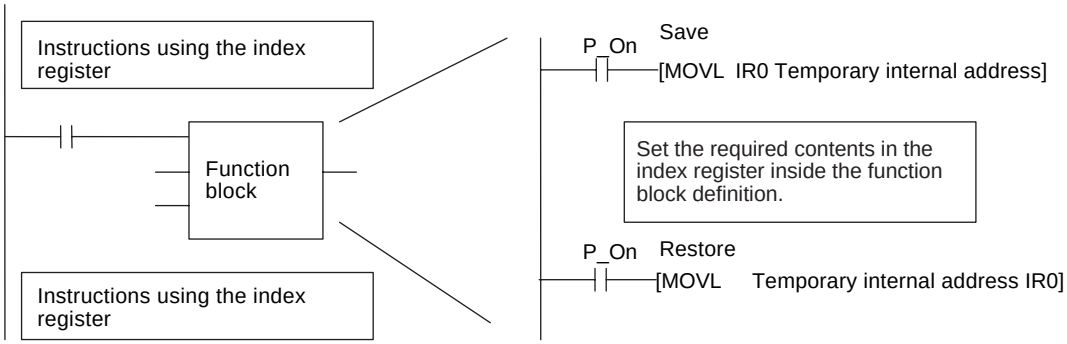
The following setting can be made from the PLC Properties Dialog Box in the CX-Programmer to control sharing index and data registers between tasks.

To share index registers between all tasks, clear the selection of the *Use IR/DRs independently per task* Check Box in the PLC Properties Dialog Box.



Additional Information

The contents of an index register used inside a function block may be corrupted when the function block is called. Always save the contents of the index register before calling the function block and then restore the contents after leaving the function block. Set the required contents in the index register inside the function block.



6-19 Data Registers

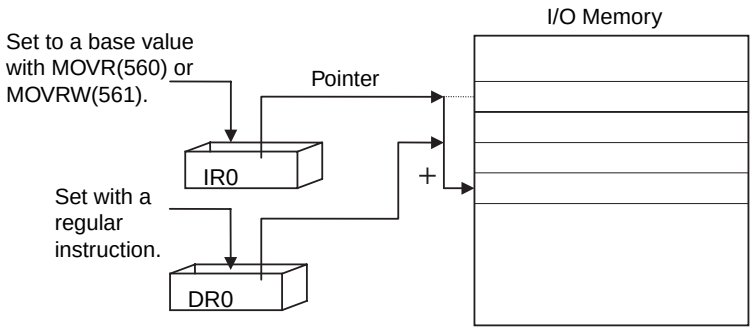
The sixteen Data Registers (DR0 to DR15) are used to offset the PLC memory addresses in Index Registers when addressing words indirectly. The Data Registers can be used to specify an offset to add to an Index Register when addressing words indirectly.

Data registers can be used independently for each task or shared between tasks.

The value in a Data Register can be added to the PLC memory address in an Index Register to specify the absolute memory address of a bit or word in I/O memory. Data Registers contain signed binary data, so the content of an Index Register can be offset to a lower or higher address.

Normal instructions can be use to store data in Data Registers.

Data Registers cannot be force-set or force-reset.



Examples

The following examples show how Data Registers are used to offset the PLC memory addresses in Index Registers.

LD DR0 ,IR0
Adds the contents of DR0 to the contents of IR0 and loads the bit at that PLC memory address.

MOV(021) #1 DR0 ,IR1
Adds the contents of DR0 to the contents of IR1 and writes #1 to that PLC memory address.

Range of Values

The contents of data registers are treated as signed binary data and thus have a range of –32,768 to 32,767. (Negative values are given as the two's complement.)

Hexadecimal content	Decimal equivalent
8000 to FFFF	–32,768 to –1
0000 to 7FFF	0 to 32,767

Data Register Initialization

The Data Registers will be cleared in the following cases:

- The operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa and the IOM Hold Bit is OFF.
- The PLC's power supply is cycled and the IOM Hold Bit is OFF or not set to be held in the PLC Setup.
- A fatal error occurs (except for one created with FALS(007)).

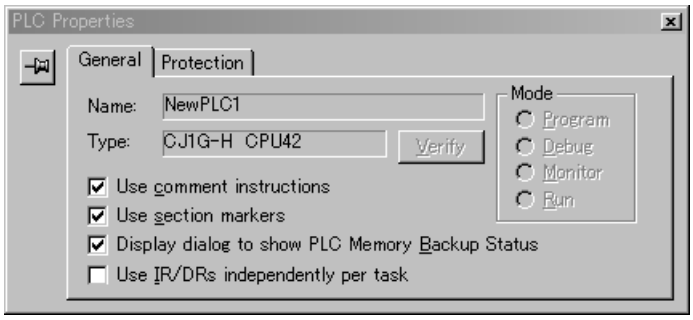
IOM Hold Bit Operation

If the IOM Hold Bit (A500.12) is ON, the Data Registers won't be cleared when a FALS error occurs or the operating mode is changed from PROGRAM mode to RUN/MONITOR mode or vice-versa.

If the IOM Hold Bit (A500.12) is ON and the IOM Hold Bit Check Box is selected in the Startup Hold Area on the Startup Tab Page in the PLC Setup, the Data Registers won't be cleared when the PLC's power supply is cycled.

Sharing Data Registers

The following setting can be made from the PLC Properties Dialog Box in the CX-Programmer to control sharing index and data registers between tasks.



Additional Information

We recommend setting the PLC properties to share index and data registers between tasks whenever index and data registers are not used in the program.

6-20 Condition Flags

The Condition Flags include the Error Flag, Carry Flag, and other flags that indicate the results of instruction execution, as well as Always ON and Always OFF Flags. In earlier PLCs, these flags were in the Auxiliary Area.

The Condition Flags are specified with global symbols, such as P_CY and P_ER, rather than addresses. These flags cannot be written directly from instructions or CX-Programmer.

All Condition Flags are cleared when the program switches tasks, so the status of the Error Flag, Access Error Flag, and other flags are maintained only in the task in which the error occurred.

The Condition Flags cannot be force-set and force-reset.

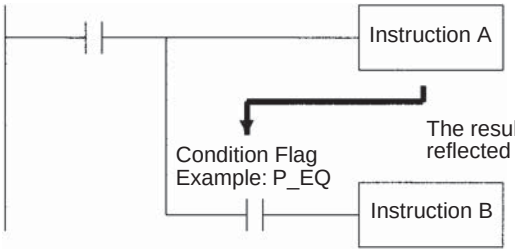
● Summary of the Condition Flags

The following table summarizes the functions of the Condition Flags.

Name	Symbol	Label	Function
Error Flag	P_ER	ER	Turned ON when the operand data in an instruction is incorrect (an instruction processing error) to indicate that an instruction ended because of an error. When the PLC Setup is set to stop operation for an instruction error (Instruction Error Operation), program execution will be stopped and the Instruction Processing Error Flag (A295.08) will be turned ON when the Error Flag is turned ON.
Access Error Flag	P_AER	AER	Turned ON when an Illegal Access Error occurs. The Illegal Access Error indicates that an instruction attempted to access an area of memory that should not be accessed. When the PLC Setup is set to stop operation for an instruction error (Instruction Error Operation), program execution will be stopped and the Instruction Processing Error Flag (A4295.10) will be turned ON when the Access Error Flag is turned ON.
Carry Flag	P_CY	CY	Turned ON when there is a carry in the result of an arithmetic operation or a “1” is shifted to the Carry Flag by a Data Shift instruction. The Carry Flag is part of the result of some Data Shift and Symbol Math instructions.
Greater Than Flag	P_GT	>	Turned ON when the first operand of a Comparison Instruction is greater than the second or a value exceeds a specified range.
Equals Flag	P_EQ	=	Turned ON when the two operands of a Comparison Instruction are equal the result of a calculation is 0.
Less Than Flag	P_LT	<	Turned ON when the first operand of a Comparison Instruction is less than the second or a value is below a specified range.
Negative Flag	P_N	N	Turned ON when the most significant bit (sign bit) of a result is ON.
Overflow Flag	P_OF	OF	Turned ON when the result of calculation overflows the capacity of the result word(s).
Underflow Flag	P_UF	UF	Turned ON when the result of calculation underflows the capacity of the result word(s).
Greater Than or Equals Flag	P_GE	>=	Turned ON when the first operand of a Comparison Instruction is greater than or equal to the second.
Not Equal Flag	P_NE	< >	Turned ON when the two operands of a Comparison Instruction are not equal.
Less Than or Equals Flag	P_LE	< =	Turned ON when the first operand of a Comparison Instruction is less than or equal to the second.
Always ON Flag	P_On	ON	Always ON.
Always OFF Flag	P_Off	OFF	Always OFF.

Using the Condition Flags

The Condition Flags are shared by all of the instructions, so their status may change often in a single cycle. Be sure to read the Condition Flags immediately after the execution of instruction, preferably in a branch from the same execution condition.



Instruction	Operand
LD	
Instruction A	
AND	P_EQ
Instruction B	

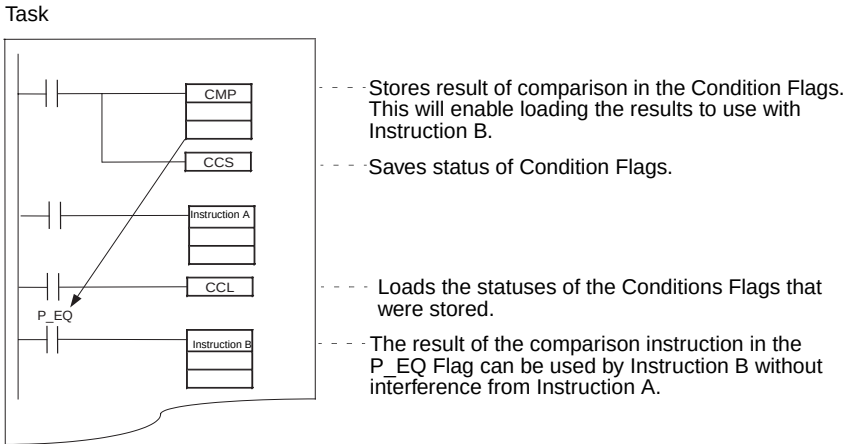
Precautions for Correct Use

- Because the Condition Flags are shared by all of the instructions, program operation can be changed from its expected course by interruption of a single task. Be sure to consider the effects of interrupts when writing the program.
- The Condition Flags are cleared when the program switches tasks, so the status of a Condition Flag cannot be passed to another task. For example the status of a flag in task 1 cannot be read in task 2.

Saving and Loading Condition Flag Status

The Condition Flag status instructions (CCS(282) and CCL(283)) can be used to save and load the status of the Condition Flags between different locations within a task (program) or between different tasks or cycles.

The following example shows how the Equals Flag is used at a different location in the same task.

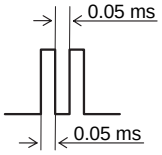
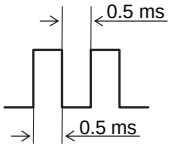
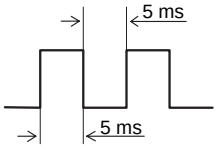
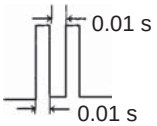
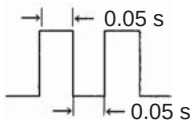
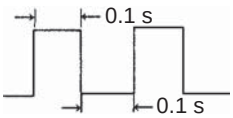
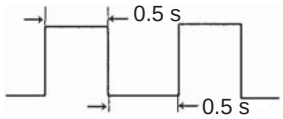
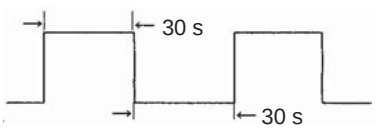


6-21 Clock Pulses

The Clock Pulses are flags that are turned ON and OFF at regular intervals by the system.

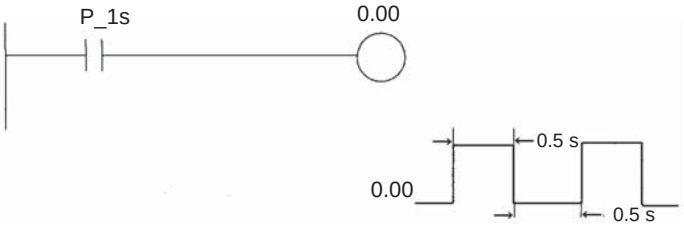
The Clock Pulses are read-only; they cannot be overwritten from instructions or the CX-Programmer. They are cleared at the start of operation.

Clock Pulses are specified using global symbols, such as P_0_1ms and P_1ms.

Name	Symbol	Name on Programming Console	Operation	
0.1 ms Clock Pulse	P_0_1ms	0.1 ms		ON for 0.05 ms OFF for 0.05 ms
1 ms Clock Pulse	P_1ms	1 ms		ON for 0.5 ms OFF for 0.5 ms
0.01 s Clock Pulse	P_0_01s	0.01 s		ON for 5 ms OFF for 5 ms
0.02 s Clock Pulse	P_0_02_s	0.02 s		ON for 0.01 s OFF for 0.01 s
0.1 s Clock Pulse	P_0_1s	0.1 s		ON for 0.05 s OFF for 0.05 s
0.2 s Clock Pulse	P_0_2s	0.2 s		ON for 0.1 s OFF for 0.1 s
1 s Clock Pulse	P_1s	1 s		ON for 0.5 s OFF for 0.5 s
1 min Clock Pulse	P_1min	1 min		ON for 30 s OFF for 30 s

Using the Clock Pulses

The following example turns CIO 0.00 ON and OFF at 0.5 s intervals.



Instruction	Operand
LD	P_1s
OUT	0.00

Clock Pulse Refreshing

The clock pulses are refreshed even during program execution. ON/OFF status may not be the same at the beginning and end of a program.

Clock Pulse Error

The maximum error in the clock pulses is 0.01% (at 25°C). For long-term, time-based control, we recommend you use the internal clock instead of the clock pulses. Be sure to allow for the error in the internal clock.

7

File Operations

This section describes the file operations that can be performed with CJ2 CPU Units.

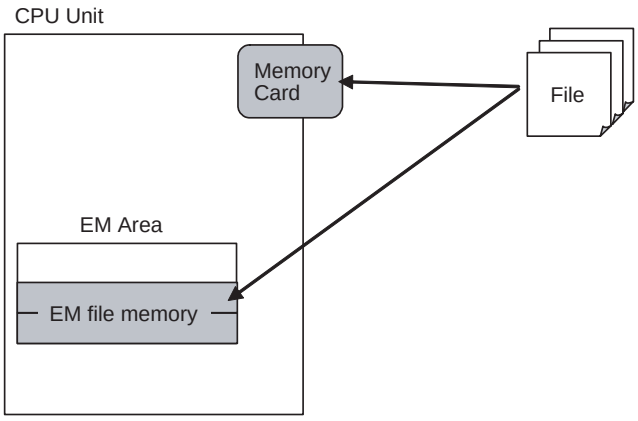
7-1 File Memory 7-2
7-1-1 Types of File Memory 7-2
7-1-2 Initializing File Memory 7-3
7-1-3 Memory Card Precautions 7-5
7-2 Types of Files Stored in File Memory 7-7
7-2-1 File Types 7-7
7-2-2 Creating and Saving Files for File Memory 7-10
7-3 File Memory Operations 7-11
7-3-1 Types of File Memory Operations 7-11
7-3-2 File Memory Operating Procedures and File Memory Files 7-13
7-3-3 Restrictions on File Use 7-19
7-3-4 File Sizes 7-20
7-3-5 Relation between Support Software and File Memory Files 7-21

7-1 File Memory

7-1-1 Types of File Memory

File memory can be used to store files in CJ-series PLCs. The two following types of file memory are used.

- Memory Cards
- A specified range in the EM Area called EM file memory



With the CJ-series PLCs, Memory Cards and a specified range of the EM Area can be used as memory to store files. Either can be used to save the entire user program, I/O memory, and parameter areas as files.

Category	Memory type	Memory capacity	Storable file types
Memory Card	Flash memory	HMC-EF183: 128 Mbytes HMC-EF283: 256 Mbytes HMC-EF583: 512 Mbytes	• Program files • Comment files • Program index files • Symbol table files • Parameter files • Unit and Board backup files (Memory Card only)
Built-in RAM (EM File Memory)	RAM	EM Area capacity up to 13 banks from bank specified in EM Area to last bank in EM Area	

7-1-2 Initializing File Memory

Memory Cards

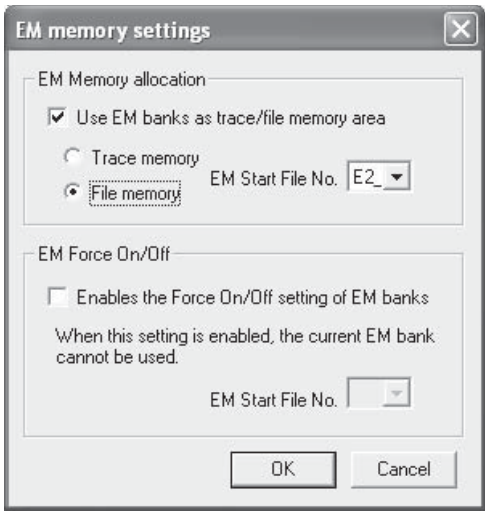
Use the Memory Card Dialog Box in the CX-Programmer to initialize Memory Cards. The HMC-EF□□□ Memory Card, however, does not need to be initialized when it is first used, because it is already initialized by default.

EM File Memory

Use the following procedure to specify the first bank in the EM file memory to be used as file memory and to perform initialization the EM file memory.

1 Select ***PLC*** – ***Memory Allocate*** – ***EM Memory Settings*** in the CX-Programmer.

2 Select *File Memory* in the EM Memory Settings Dialog Box, and then set the EM start File No.



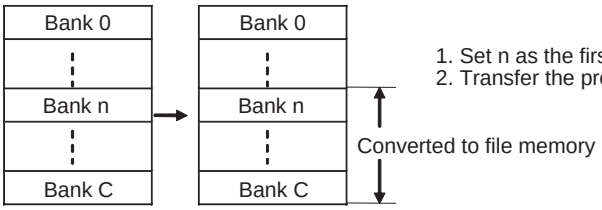
Additional Information

- The EM Area cannot be used for both file memory and trace memory at the same time. Use only one of these functions.
- Force-setting/resetting is not possible for EM banks that are used as trace/file memory.

3 Connect online to the CPU Unit and transfer the user program. It will then be possible to use the EM file memory.

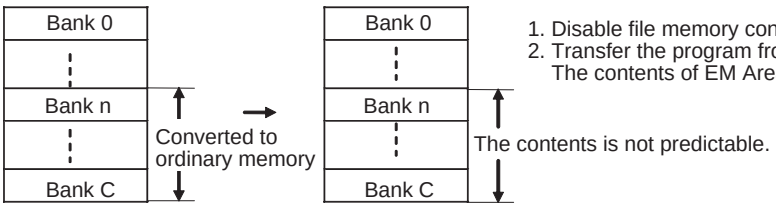
Changing EM File Memory Settings

The following figures shows converting the EM Area from a specified bank to the last back to file memory.



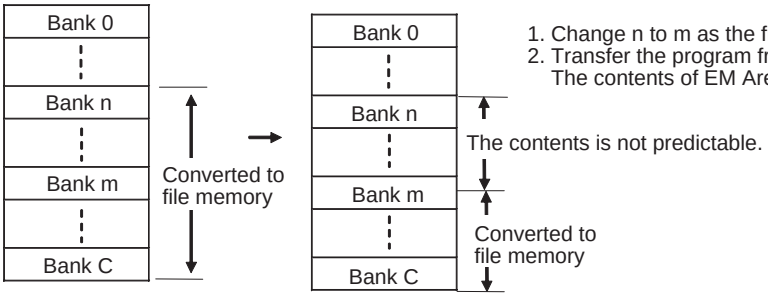
1. Set n as the first EM file memory bank in the PLC Setup.
2. Transfer the program from the CX-Programmer.

The following figures shows restoring EM Area banks used for file memory restored to ordinary EM Area memory.



1. Disable file memory conversion in the PLC Setup.
2. Transfer the program from the CX-Programmer.
   The contents of EM Area after bank n will not be predictable.

The start bank number for file memory can be changed.



1. Change n to m as the first EM file memory bank in the PLC Setup.
2. Transfer the program from the CX-Programmer.
   The contents of EM Area for banks n to m will not be predictable.

7-1-3 Memory Card Precautions

Confirm the following items before using a Memory Card.

Format

Memory Cards are formatted before shipping. There is no need to format them after purchase. To format them once they have been used, always do so in the CPU Unit using the CX-Programmer. If a Memory Card is formatted directly in a notebook computer or other computer, the CPU Unit may not recognize the Memory Card. If this occurs, you will not be able to use the Memory Card even if it is reformatted in the CPU Unit.

Number of Files in Root Directory

There is a limit to the number of files that can be placed in the root directory of a Memory Card (just as there is a limit for a hard disk). Although the limit depends on the type and format of the Memory Card, it will be between 128 and 512 files. When using applications that write log files or other files at a specific interval, write the files to a subdirectory rather than to the root directory. Subdirectories can be created on a computer or by using the CMND(490) instruction.

Number of Writes

Generally speaking, there is a limit to the number of write operations that can be performed for a flash memory. For the Memory Cards, a limit of 100,000 write operations has been set for warranty purposes. For example, if the Memory Card is written to every 10 minutes, over 100,000 write operations will be performed within 2 years.

Minimum File Size

If many small files, such as ones containing only a few words of DM Area data, are stored on the Memory Card, it will not be possible to use the complete capacity of the Memory Card. For example, if a Memory Card with an allocation unit size of 4,096 bytes is used, at least 4,096 bytes of memory will be used for each file regardless of how small the file is. If you save 10 words of DM Area data to the Memory Card, 4,096 bytes of memory will be used even though the actual file size is only 68 bytes. Using files of such a small size greatly reduces the utility rate of the Memory Card. If the allocation unit size is reduced to increase the utility rate, however, the access speed will be reduced.

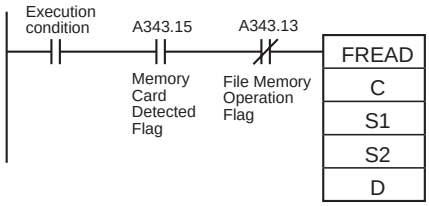
The allocation unit size of the Memory Card can be checked from a DOS prompt using CHKDSK. The specific procedure is omitted here. Refer to general computer references for more information on allocation unit sizes.

Memory Card Recognition Time

Several seconds is normally required for the CPU Unit to recognize the Memory Card after it is inserted. The required time depends on the PLC's cycle time, the Memory Card Capacity, the number of files stored on the Memory Card, and other factors. For example, the recognition time for an HMC-EF183 Memory Card will be roughly 8 s if the cycle time is 0.4 ms and all PLC Setup parameters are set to the default values

Accessing the Memory Card

- Never turn OFF the power supply to the CPU Unit when the BUSY indicator is lit (i.e., indicator showing that the Memory Card is being accessed). The Memory Card may become unusable if this is done.
- Never remove the Memory Card from the CPU Unit when the BUSY indicator is lit. Press the Memory Card power OFF button and wait for the BUSY indicator to go out before removing the Memory Card. The Memory Card may become unusable if this is not done.
- A few seconds will be required for the CPU Unit to recognize the Memory Card after it is inserted. When accessing a Memory Card immediately after turning ON the power supply or inserting the Memory Card, program an NC condition for the Memory Card Recognized Flag (A343.15) as an input condition, as shown below.



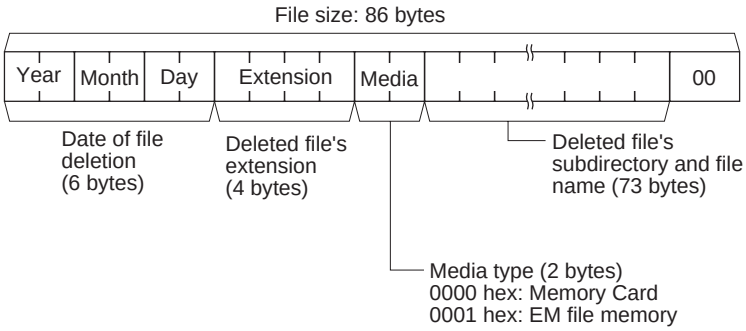
Power Interruptions While Accessing File Memory

If a power interruption occurs while the CPU is accessing file memory (the Memory Card or EM file memory) the contents of the Memory Card may not be accurate. The file being updated may not be overwritten correctly and, in some cases, the Memory Card itself may be damaged.

The affected file will be deleted automatically by the system the next time that power is turned ON. The corresponding File Deletion Notification Flag (A395.06 for the Memory Card, A395.07 for EM file memory) will be turned ON. The flag will be turned OFF the next time that the power is turned OFF.

When a file is deleted, a deletion log file (DELETE.TMP) will be created in the root directory of the Memory Card or EM file memory. The deletion log file can be read with a binary editor to check the following information: The date that the file was deleted, the type of file memory (media) that existed, the subdirectory, file name, and extension. When necessary, recreate or recopy the deleted file.

The following diagram shows the structure of the deletion log file.



7-2 Types of Files Stored in File Memory

File memory can be used for the following:

- Program/network symbol files
- Parameter files
- Data files
- Symbol table files (except network symbols)
- Comment files
- Program index files
- Unit backup files

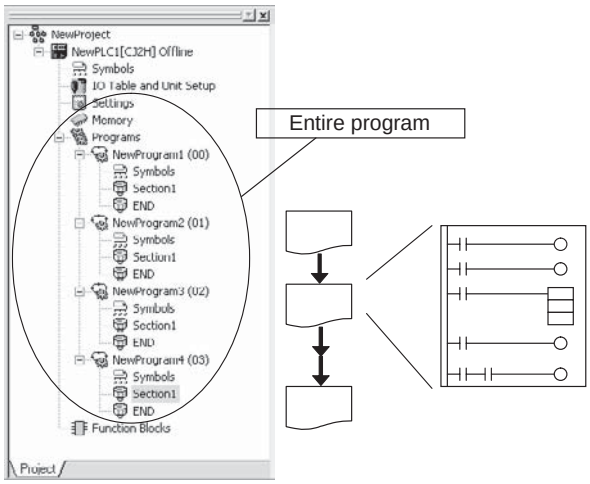
7-2-1 File Types

Program/Network Symbol File

The program/network symbol file contains the CPU Unit's user program (the programs in the cyclic tasks and interrupt tasks) and network symbols (i.e., network symbols in global symbol tables)*.

This file also contains each program's properties.

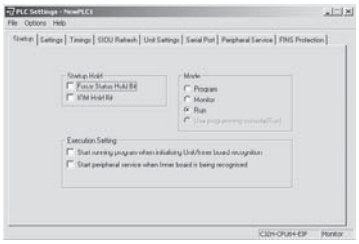
* Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.



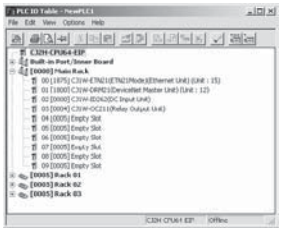
Parameter File

The parameter file contains the CPU Unit's internal Parameter Area data. The Parameter Area data includes the PLC name, PLC Setup, I/O tables, CPU Bus Unit settings (including the data link parameters), and routing tables.

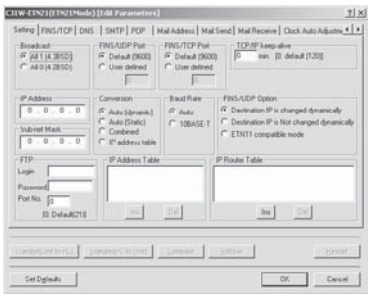
PLC Setup



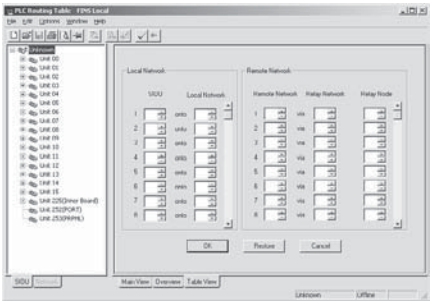
I/O tables



CPU Bus Unit Settings

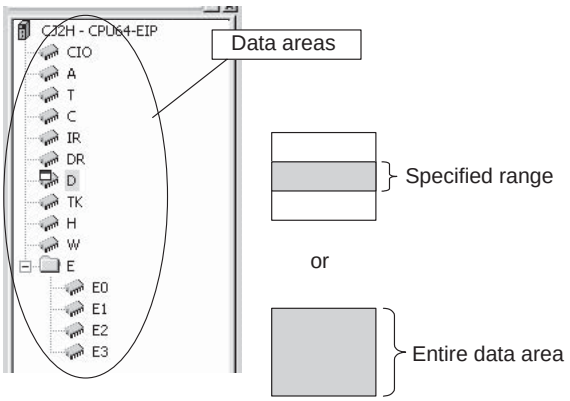


Routing tables



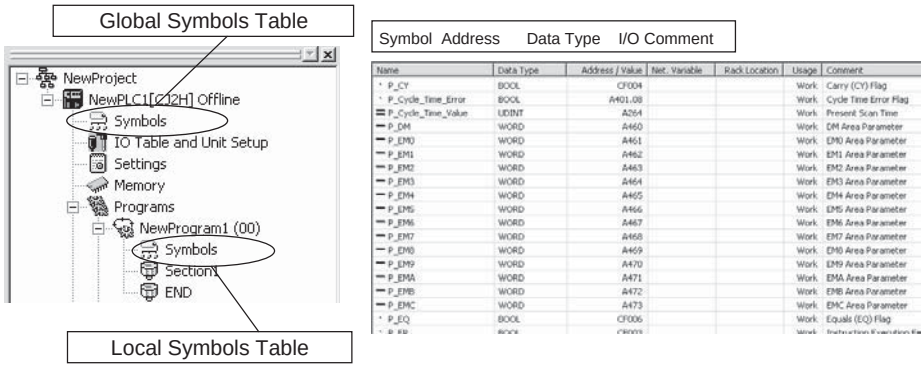
Data File

The data file contains the data of one I/O memory data area, in word (16-bit) units. It is possible to store all of the data in the data area or just a specified range of addresses. Any one of the following 6 data areas can be stored: the CIO, Holding, Work, Auxiliary, DM, or EM Area.



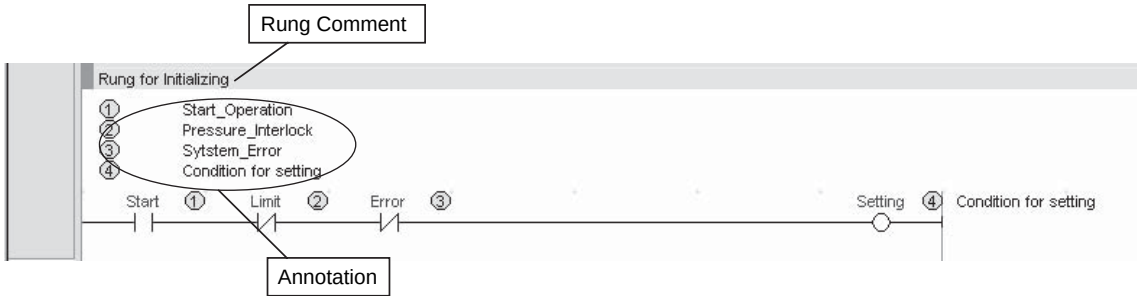
Symbols Table File (Except Network Symbols)

The symbol table file (except network symbols) contains the symbol table information used in the CX-Programmer as well as the automatically allocated PLC addresses.



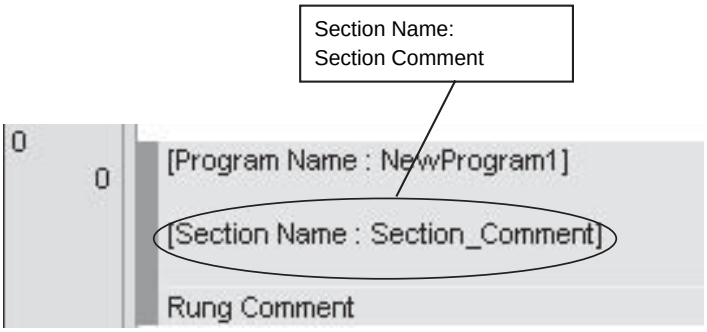
Comment File

The comment file contains the comment information used in the CX-Programmer.



Program Index File

The program index file contains the section information used in the CX-Programmer.



Unit Backup File

The Unit backup file contains the internal data of a PLC Unit, which is used by the simple backup function. These files are created when the simple backup operation is executed. Internal data is stored for each Unit.

Example:

DeviceNet Units: Device parameters

Serial Communications Units: Protocol macro data

7-2-2 Creating and Saving Files for File Memory

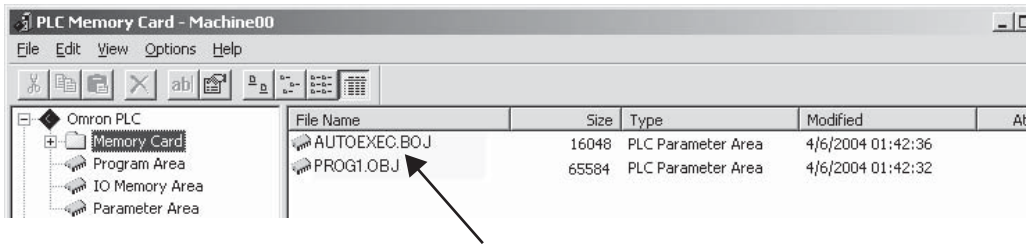
User Program, I/O Memory, and Parameter Area Data

Transfer the data using the PLC Memory Card Dialog Box in the CX-Programmer.

1 Install a Memory Card that has already been initialized into the CPU Unit or initialize the EM file memory. For information on initializing the EM file memory, refer to *7-1-2 Initializing File Memory*.

2 Place the CX-Programmer online with the CPU Unit.

3 Select the transfer destination (i.e., Memory Card or EM file memory) in the PLC Memory Card Dialog Box, and then select ***Transfer*** from the File Menu.
The data in the CPU Unit (i.e., user program, I/O memory, and parameter areas) will be saved as a file in the Memory Card or EM file memory.



Program Files Transferred Automatically at Startup

Symbol Tables, Comments, and Program Index

Transfer the data using the CX-Programmer.

1 Insert a formatted Memory Card into the CPU Unit or format EM file memory. For information on initializing EM file memory, refer to *7-1-2 Initializing File Memory*.

2 Place the CX-Programmer online with the CPU Unit.

3 Select ***Transfer*** and then ***To PLC*** or ***From PLC*** from the PLC Menu.

4 Select either **Symbols** or **Comments** as the data to transfer.

Note If a Memory Card is installed in the CPU Unit, data can be transferred only with the Memory Card. (It will not be possible with EM file memory.)

7-3 File Memory Operations

7-3-1 Types of File Memory Operations

The following can be performed to use file memory:

- Procedures from the CX-Programmer
- Automatic transfer at startup
- Simple backup function
- FREAD(700) and FWRIT(701) instructions
- Replacement of the entire program using Auxiliary Area control bits
- FINS commands

Procedures from the CX-Programmer

Refer to the CX-Programmer operation manual for details on the following procedures.

● Transferring Data in the Memory Card Dialog Box of the CX-Programmer

The following files can be created, read, and written.

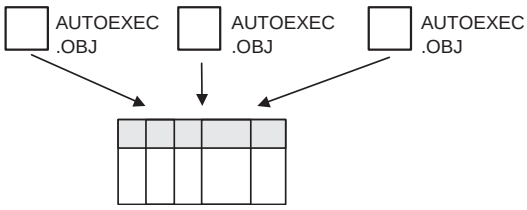
- Files automatically transferred at startup (program/network symbol files, data files, parameter files)
- General-purpose files (program/network symbol files, data files, parameter files)
- CX-Programmer files (symbol tables, comment files, program index files)

● Transferring Data Online/Offline

General-purpose files (i.e., program/network symbol files, data files, and parameter files) can be transferred online to the PLC or offline to a computer memory device.

Files Transferred Automatically at Startup

Changeover programs, parameters, and I/O memory can be stored in a Memory Card and read from the Memory Card at startup. (Refer to *Section 10 CPU Unit Functions* for details.)



Simple Backup Operation

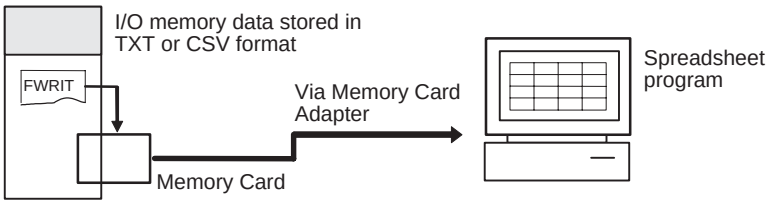
The simple backup operation enables backing up and restoring all PLC data, including data in the CPU Unit, Special I/O Units, and CPU Bus Units, to and from a Memory Card without requiring a Programming Device. (Refer to the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472) for details.)

Backup Restore Operation

All PLC data, including data in the CPU Unit, Special I/O Units, and CPU Bus Units, can be backed up and restored on a computer by using PLC Backup Tool. (Refer to the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472) for details.)

FWRIT(701)/FREAD(700) Instructions

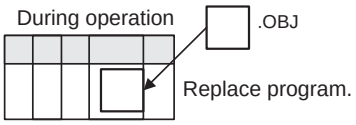
I/O memory data can be saved on the Memory Card or in EM file memory in the text or CSV format by using the FWRIT(701) instruction in the user program. It can then be transferred to a computer via a Memory Card Adapter and edited with a spreadsheet program.



Conversely, data such as Special I/O Unit settings can be created with a spreadsheet program in text or CSV format, stored on a Memory Card, and read to the CPU Unit by using the FREAD(700) instruction. Refer to the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474) for details

Replacing the User Program during Operation Using an Auxiliary Area Control Bit

The entire user program can be replaced during operation from the program itself (without a Programming Device) using an Auxiliary Area control bit. (Refer to *Section 10 CPU Unit Functions* for details.)



FINS Commands

File memory can be manipulated by using a FINS command for file memory. A FINS command can be sent from a computer connected using the Host Link protocol. Another PLC on a network can send a FINS command using CMND(490) or the local PLC can send a FINS command using CMND(490) in the user program. (Refer to the *Communications Command Reference Manual* (Cat. No. W474) for details.)

7-3-2 File Memory Operating Procedures and File Memory Files

The following table summarizes the file memory files that can be manipulated for each type of file memory operation.

Read: Transfers files from file memory to the CPU Unit.

Write: Transfers files from the CPU Unit to file memory.

OK: Possible; ---: Not possible

Operating procedure	Medium	Description	File types				
			Program/network symbol files	Data files	Parameter Area data	Symbol tables, comment files, program index files	Unit backup files
Procedures from CX-Programmer	Memory Card, EM file memory, or computer memory device	Create	OK (general-purpose files) OK (files transferred automatically at startup)			OK (CX-Programmer files)	---
		Read					
		Write					
		Other operations					
Automatic transfer at startup	Memory Card	Read	OK (files transferred automatically at startup)			---	
		Write	---				
Simple backup function	Memory Card	Read	OK (simple backup files)				
		Write					
Backup or restore using PLC Backup Tool	Computer memory device only	Read	OK (PLC backup files)				
		Write					
FREAD(700) and FWRIT(701) instructions	Memory Card or EM file memory	Read data from one file.	---	OK (general-purpose files)	---	---	
		Write data to one file.					
Auxiliary Area control bit operation replaces the entire program during operation.	Memory Card	Read	OK (general-purpose files)	---			
FINS command	Memory Card or EM file memory	Read	OK (general-purpose files)			---	
		Write					
		Other operations					

Files Automatically Transferred at Startup

These files are for batch-reading user program/network symbols, parameter data, and I/O memory at startup. The file name is fixed as AUTOEXEC or REPLACE. For details, refer to *10-3-5 Automatic Transfer at Startup*.

Backup Files

These files are saved in the Memory Card and back up all PLC data by using the DIP switch on the front of the CPU Unit or the Memory Card's power supply switch. The file name is fixed as BACKUP□□. For details, refer to information on the *8-3 Simple Backup* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).

PLC Backup Tool Files

These files back up all PLC data and are created using backup and restore operations with the PLC Backup Tool. For details, refer to *8-2 Using a Computer to Back Up Data* in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).

CX-Programmer Files

These files are created by the CX-Programmer. The file names created in file memory are fixed. When a project is transferred, any one of the following memory options can be selected as the transfer destination for these CX-Programmer files.

- Comment memory (in the CPU Unit's flash memory)
- Memory Card
- EM file memory

Type	Name	Description	Explanation
Symbol table files (except network symbols)	SYMBOLS.SYM	Global symbol tables (except network symbols) and local symbol tables	• Includes the variables in the CX-Programmer's global/local symbol tables, addresses, data types, and I/O comment information. Network symbols are not included. • Also includes the data set in the CX-Programmer's automatic PLC address allocation.
Comment files	COMMENTS.CMT	Rung comments and comments (annotations)	Contains the CX-Programmer's rung comment and comment information.
Program index files	PROGRAM.IDX	Section names, section comments	Contains the CX-Programmer's section delimiter information, although the delimiter locations depend on the section delimiter instructions in the program.

Note With the CX-Programmer, the symbol table files and comment files in the above table (except network symbols) can be transferred online between the CX-Programmer and personal computer RAM and between the personal computer RAM and the memory storage device.

When a project is transferred from the CX-Programmer to the CPU Unit, symbol tables, comment files, and program index files can be stored in the flash memory built into the CPU Unit. When the simple backup operation is performed, symbol tables, comment files, and program index files stored in the comment memory will be automatically created as backup files and stored in the Memory Card.

General-purpose Files

Read, write and other operations can be performed on these files using the CX-Programmer, FINS commands, instructions, and Auxiliary Area bits.

The file names are user-defined.

File type	File name*1	Extension	Content	Description	
Program/network symbol file	********	.OBJ	• User program • Network symbols*2	Programs and network symbols in cyclic tasks and interrupt tasks of the CPU Unit.	
Parameter file	********	.STD	• PLC Setup • Registered I/O tables • Routing tables • CPU Bus Unit Setup Area	• Parameter area data in the CPU Unit • The user does not need to differentiate the data in the file.	
Data file	********	.IOM	Specified range of I/O memory	• Data in word (16-bit) increments in one area from the start word to the end word. • The area can be CIO, HR (Holding Area), WR (Work Area), AR (Auxiliary Area), DM (Data Memory), or EM (Extended Memory).	Binary data format
		.TXT			Text format (no delimiter or tabs)*3
		.CSV			CSV format (comma delimited)

*1 For ********, set eight ASCII characters or less.

*2 Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.

*3 The text and CSV data files can be read and written only by using FWRIT(701) and FREAD(700) instructions. They cannot be read or written from the CX-Programmer.

● General-purpose Data Files

Types of Data File

There are three kinds of general-purpose data files, with filename extensions IOM, TXT, and CSV.

Extension	Contents	Purpose
.IOM	Binary format CS/CJ-series data format	I/O memory backup
.TXT*1	In these data formats, 1-word or 2-word fields in I/O memory are converted to ASCII data. Records can be delimited with carriage returns.	Exchanging data with spreadsheet software
.CSV		

*1 Reading and Writing TXT and CSV Data Files:
TXT and CSV data files can be read and written with FREAD(700) and FWRIT(701) only.

The following six data formats are used in text and CSV files.

Extension	Data format	Contents	
		I/O memory size per field	Delimiter
.TXT*1	Non-delimited words	1 word	None
	Non-delimited double words	2 words	None
	Tab-delimited words	1 word	Tab code
	Tab-delimited double words	2 words	Tab code
.CSV*1	Comma-delimited words	1 word	Comma
	Comma-delimited double words	2 words	Comma

*1 Precautions on Characters:
Data cannot be written to I/O memory properly if the TXT or CSV file contains characters other than hexadecimal characters (0 to 9, A to F, or a to f.)

- Precautions on Field Size:
  When words are being used, data cannot be written to I/O memory properly if the TXT or CSV file contains fields that are not 4-digit hexadecimal. Likewise, when double words are being used, data cannot be written properly if the file contains fields that are not 8-digit hexadecimal.
- Storage Order:
  When words are being used, I/O memory data is converted to ASCII and stored in one-word fields in order from the lowest to the highest I/O memory address.
  When double words are being used, I/O memory data is converted to ASCII and stored in two-word fields in order from the lowest to the highest I/O memory address. (Within the two-word fields, the higher-address word is stored first and the lower-address word is stored second.)
- Delimiters:
  When there are no delimiters, the fields are packed consecutively and then stored. When delimited by commas, commas are inserted between fields before they are stored. When delimited by tabs, tab codes are inserted between fields before they are stored.
  When delimiters (commas or tabs) are specified in FREAD(700), the data is read as delimited data with one-word delimiters (commas or tabs).
- Carriage Returns:
  Data is packed consecutively when carriage returns are not used.
  When carriage returns are used, a carriage return code is inserted after the specified number of fields. An offset from the beginning of the file (starting read word or starting write word) cannot be specified in the FREAD(700)/FWRIT(701) instructions if carriage returns are used in the file.
- Number of Fields:
  The overall amount of data in the file depends upon the number fields (number of write items) specified in the FWRIT(701) instruction and the number of words/field.
  The size is specified with the number of fields.
  With word data, 1 word in I/O memory = 1 field
  With long word data, 2 words in I/O memory = 1 field.

Data File Names

Data files do not contain information indicating what data is stored, i.e., what memory area is stored. Be sure to give file names that indicate the contents, as shown in the examples below, to aid in file management.
Examples: D00100.IOM, CIO0020.IOM
Data from the beginning of the file will be written starting at the address specified in I/O memory even if the data originally written to the data file (IOM, TXT, or CSV) is not from the same area. For example, if CIO data in a file is written to the DM Area from a Programming Device, the data will be read to the DM Area of the CPU Unit without any indication that the area is different.

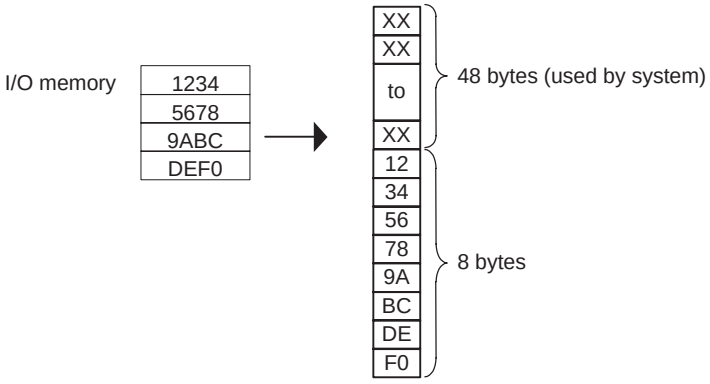
Additional Information

Data files with the TXT and CSV format contain hexadecimal (0 to 9, A to F) data that allows the I/O memory numerical data to be exchanged with spreadsheet programs.

ASCII data in I/O memory can be saved as a text file by using the TWRIT(704) (WRITE TEXT FILE) ladder programming instruction or the Memory Card processing function WRITE_TEXT ( ) in ST language.

Additional Information

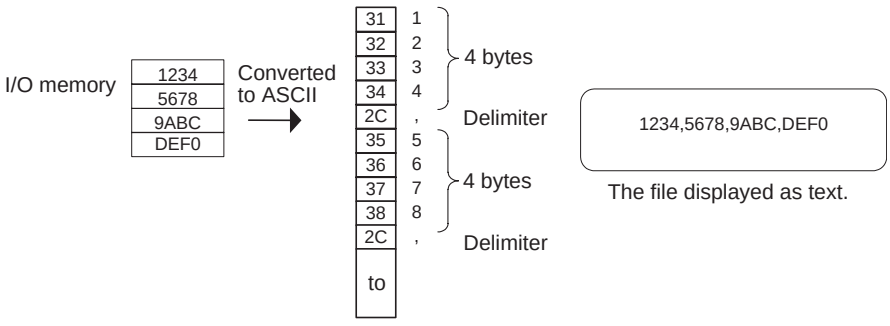
The following illustration shows the binary data structure of a data file (ABC.IOM) containing four words from I/O memory: 1234 Hex, 5678 Hex, 9ABC Hex, and DEF0 Hex. The user, however, does not have to consider the data format in normal operations.



Contents of ABC.IOM

- Structure of CSV and TXT Data Files with Single-word Fields

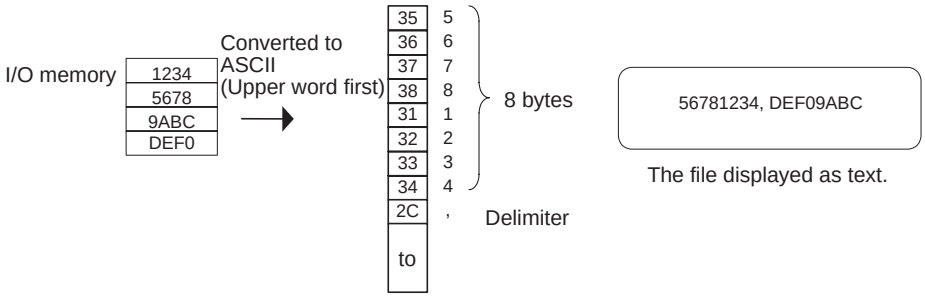
The following illustration shows the data structure of a CSV data file (ABC.CSV) with single-word fields containing four words from I/O memory: 1234 Hex, 5678 Hex, 9ABC Hex, and DEF0 Hex.



Contents of ABC.CSV

- Structure of CSV and TXT Data Files with Double-word Fields

The following illustration shows the data structure of a CSV data file (ABC.CSV) with double-word fields containing four words from I/O memory: 1234 Hex, 5678 Hex, 9ABC Hex, and DEF0 Hex.



Contents of ABC.CSV

Additional Information

Creating Data Files with Spreadsheet Software

Use the following procedure to create TXT and CSV data files with spreadsheet software such as Microsoft Excel.

- Set the cell contents to characters.
- Input 4 characters in each cell if single-word fields are being used or 8 characters if double-word fields are being used. For example, if single-word fields are being used input 000A, not just A.
- Be sure to input only hexadecimal characters (0 to 9, A to F, or a to f) in the cells. Other characters and codes cannot be used.

When you want to store hexadecimal digits in I/O memory, it is helpful to convert the spreadsheet's decimal inputs to hexadecimal. Use the following procedure to convert to hexadecimal.

(1) Select *Add-Ins...* **from the Tools Menu.**

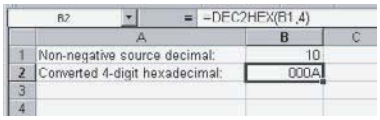
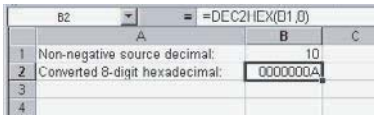
(2) Select Analysis ToolPak **in the Add-Ins Menu.**

(3) Select *Function* **from the Insert Menu at the cell where the function will be used.**

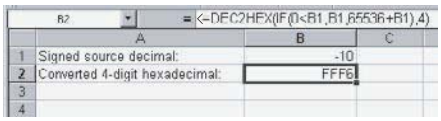
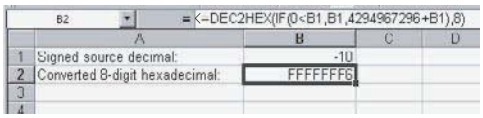
(4) Select *DEC2HEX (number, digits)* **from *Engineering* in the *Category* Field.**

(5) When converting to 4-digit hexadecimal, input the following at the number variable: IF(0<=cell location,cell location,65535+cell location)
When converting to 8-digit hexadecimal, input the following at the number variable: IF(0<=cell location,cell location,4294967296+cell location)

Example 1: Inputting Non-negative Decimal Values

Item	Converting unsigned decimal to 4-digit hexadecimal	Converting unsigned decimal to 8-digit hexadecimal
Function used	DEC2HEX(*cell_location,4*)	DEC2HEX(*cell_location,8*)
Example	Input 10 in decimal and convert to 000A in 4-digit hexadecimal. 	Input 10 in decimal and convert to 0000000A in 8-digit hexadecimal. 

Example 2: Inputting Signed Decimal Values

Item	Converting unsigned decimal to 4-digit hexa-decimal	Converting unsigned decimal to 8-digit hexa-decimal
Function used	DEC2HEX(IF(0<=*cell_location,cell_location,65536+cell_location*),4)	DEC2HEX(IF(0<=*cell_location,cell_location, 4294967296+cell_location*),8)
Example	Input -10 in decimal and convert to FFF6 in 4-digit hexadecimal.  	Input -10 in decimal and convert to FFFFFFF6 in 8-digit hexadecimal. 

7-3-3 Restrictions on File Use

Files are formatted in DOS format, and therefore can be used as regular files on a Windows computer.

● File Names

Files are identified by file names and extensions, as shown in the following table. A file name is written using the following characters: Letters a to z, A to Z, numbers 0 to 9, !, &, $, #,′, {, }, -, ^, (,), and _

The following characters cannot be used in file names: ,, ., /, \, ?, *, ", :, ;, <, >, =, +, space, and 2-byte characters.

The filename extensions depend upon the type of file being stored.

● Directories

The CS/CJ-series CPU Units can access files located in subdirectories. Specify the directory location in file memory where the file is stored. Directories can be specified up to 5 subdirectories deep (counting the root directory), unless a Programming Console is being used. The maximum length of a directory path is 65 characters. When creating a Memory Card subdirectory with an operating system such as Windows, do not exceed the maximum subdirectory depth (5 subdirectories).

7-3-4 File Sizes

The size of files in bytes can be calculated with the equations in the following table.

File type	File size
Data files (.IOM)	(Number of words × 2) + 48 bytes Example: Entire DM Area (D0 to D32767) (32,768 words × 2) + 48 = 65,584 bytes
Data files (.TXT or .CSV)	The file size depends upon the number of delimiters and carriage returns being used. The delimiter code is one byte and the carriage return code is two bytes. Example 1: Non-delimited words, no carriage return 123456789ABCDEF012345678 occupies 24 bytes. Example 2: Delimited words, carriage return every 2 fields 1234,5678↵ 9ABC,DEF0↵ 1234,5678↵ occupies 33 bytes. Example 3: Delimited double words, carriage return every 2 fields 56781234,DEF01234↵ 56781234↵ occupies 29 bytes.
Program files (.OBJ)	(Number of steps used × 4) + 48 bytes*1
Parameter files (.STD)	16,048 bytes

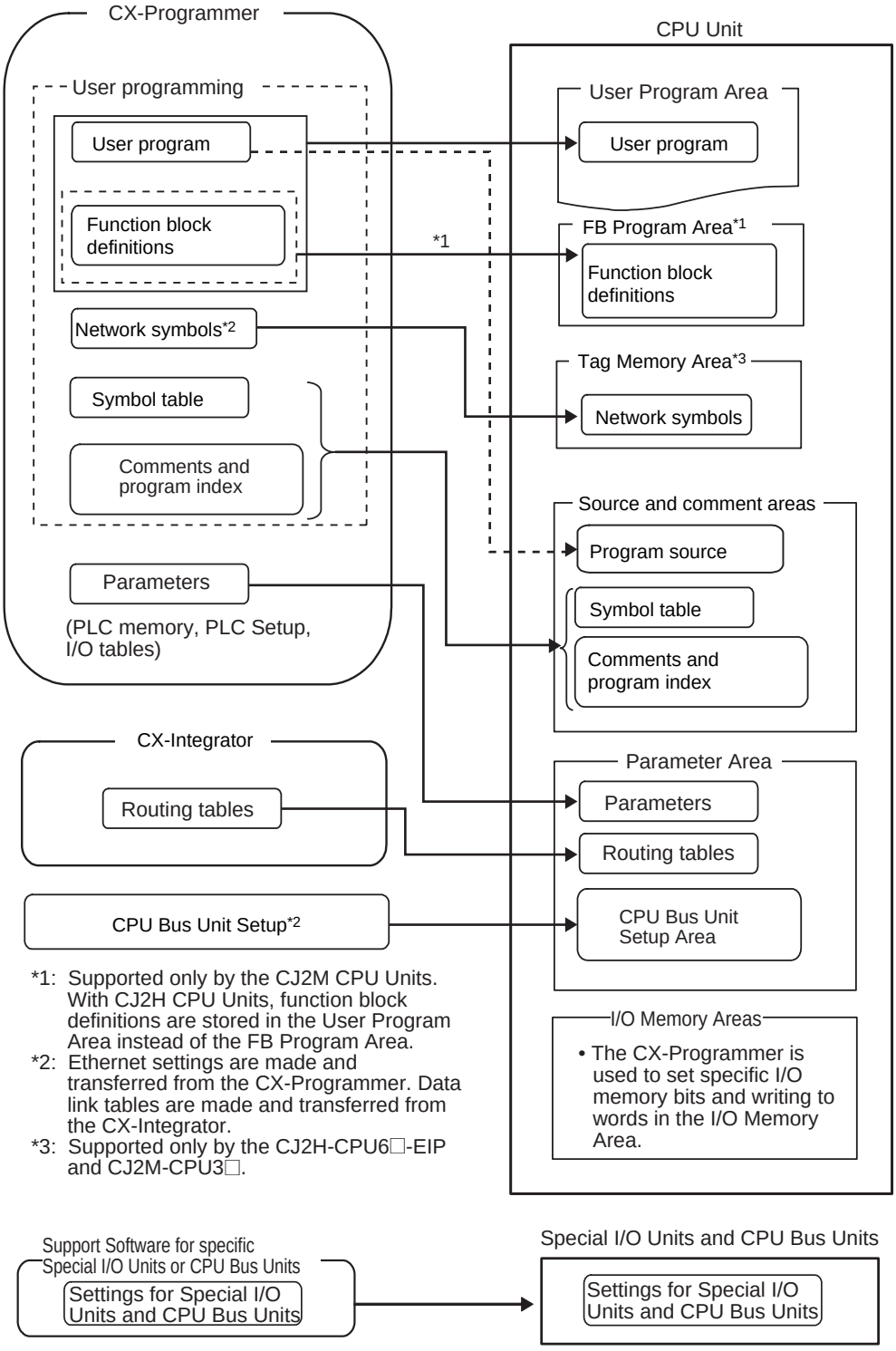
*1 Calculate the number of steps in the program file by subtracting the available UM steps from the total UM steps. These values are shown in the CX-Programmer's Cross-Reference Report. Refer to the *CX-Programmer Operation Manual* for details.

7-3-5 Relation between Support Software and File Memory Files

The following table gives the relation between the files for each Support Software package and the file memory file.

Support Software	Support Software file	Contents	File memory file	Storage area
CX-Programmer	CX-Programmer project file (extension:.cxp)	User program/network symbols	Program/network symbol file (extension: OBJ)	CPU Unit
		Function blocks		
		Symbol tables (including symbol names and address correspondence, I/O comments) Note: Excluding network symbols.	Symbol tables (except network symbols) (SYMBOLS.SYM)	
		Comments (row comments and annotations except for I/O comments)	Comment files (COMMENTS.CMT)	
		PLC name	Parameter files (extension: STD)*1	
		PLC Setup		
		Registered I/O tables		
		Ethernet Unit settings		
	Data link table files (.cl2)	Controller Link data link tables		
	CX-Programmer project file (extension: .cxp) Unit settings (XML file): CPS contents	PLC memory (I/O memory) (Allocated I/O memory and allocated DM Area words follow.)	Data files (extension: IOM)	
		Allocated I/O memory and allocated DM Area words		
CX-Integrator	Routing table files (extension: .rtg)	Routing tables	Parameter files (extension: STD)*1	
FL-net Support Software	Specific files (.csv)	FL-net settings		
Support Software for specific Special I/O Unit or CPU Bus Unit	Files created by the specific Support Software for Special I/O Units and CPU Bus Units	Data not included in the above files	Unit backup files (BACKUP□□.PRM)	Special I/O Units and CPU Bus Units

*1 Files created using a Support Software application will be created as one file in the file memory.


CX-Programmer
User programming
User program
Function block definitions
Network symbols*2
Symbol table
Comments and program index
Parameters
(PLC memory, PLC Setup, I/O tables)
CX-Integrator
Routing tables
CPU Bus Unit Setup*2
*1
CPU Unit
User Program Area
User program
FB Program Area*1
Function block definitions
Tag Memory Area*3
Network symbols
Source and comment areas
Program source
Symbol table
Comments and program index
Parameter Area
Parameters
Routing tables
CPU Bus Unit Setup Area
I/O Memory Areas
• The CX-Programmer is used to set specific I/O memory bits and writing to words in the I/O Memory Area.
*1: Supported only by the CJ2M CPU Units. With CJ2H CPU Units, function block definitions are stored in the User Program Area instead of the FB Program Area.
*2: Ethernet settings are made and transferred from the CX-Programmer. Data link tables are made and transferred from the CX-Integrator.
*3: Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.
Support Software for specific Special I/O Units or CPU Bus Units
Settings for Special I/O Units and CPU Bus Units
Special I/O Units and CPU Bus Units
Settings for Special I/O Units and CPU Bus Units

8

I/O Allocations and Unit Settings

The section describes how memory is allocated to Basic I/O Units, Special I/O Units, and CPU Bus Units and how to set the Special I/O Units and CPU Bus Units.

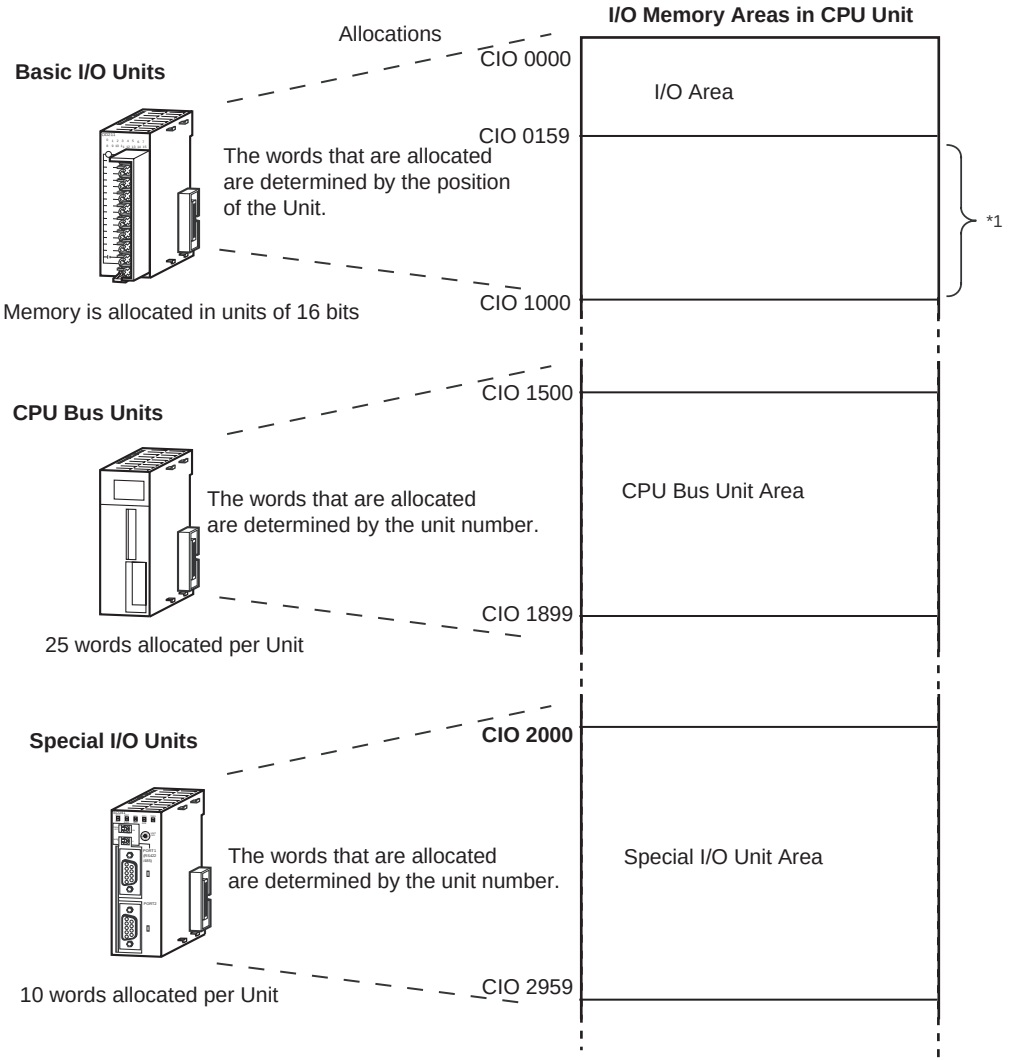
8-1 I/O Allocations 8-2
- 8-1-1 I/O Allocations 8-2
- 8-1-2 Automatic Allocation 8-5
- 8-1-3 Manual Allocation 8-9
- 8-1-4 I/O Table Errors and Precautions 8-17

8-2 Setting CPU Bus Units and Special I/O Units 8-20
- 8-2-1 Setting Parameters 8-20
- 8-2-2 Data Exchange 8-24

8-1 I/O Allocations

8-1-1 I/O Allocations

Allocating words in I/O memory in the CPU Unit for exchanging data between the CPU Unit and other Units is called I/O allocation. Memory is allocated differently to Basic I/O Units, Special I/O Units, and CPU Bus Units.



*1 The first word on each Rack can be set from the CX-Programmer to an address between CIO 0000 and CIO 0999 (default setting: consecutive allocation from CIO 0000).

I/O Allocation Methods

Use one of the following two methods to allocate I/O.

- Automatic Allocation (Default)
  When the Unit power supply is turned ON, I/O will be allocated according to the status of the connected Units (e.g., the slot positions and unit numbers).
- Manual Allocation (I/O Table Creation)
  You can create I/O tables using the CX-Programmer to specify the desired allocations. If I/O tables are created, an error will occur if the I/O tables do not match the status of connected Units when the CPU Unit Power Supply is turned ON.

The following table shows the differences between these two methods. Select the method according to the purpose and needs of the system.

Allocation method / Item	Automatic allocation	Manual allocation	
		Creating I/O tables with Units connected	Creating I/O tables with user-set allocations
Purpose	• Use this method when checking for incorrect Units is not required. • Use this method when user-set allocations are not required (including reserving unused words).	Use this method to check for incorrect Units.	• Use this method to create user-set allocations (including reserving unused words). • Use this method to allocate I/O without the actual I/O Units (i.e., offline).
Creating I/O tables	Not required.	Required.	
Verifying the connected Units against the I/O tables	Unit not verified.	Unit verified. (Operation cannot be started if the connected Units do not agree with the I/O tables.)	
Operation used to allocate I/O	Allocated automatically.	Allocated manually.	
CX-Programmer	Not required.	Required	
Connected Units required to create I/O tables	Required.	Required	Not required.
Allocation method	Automatic allocation (in order from the CPU Unit)		Manual allocation (The first address can be specified for each group.)

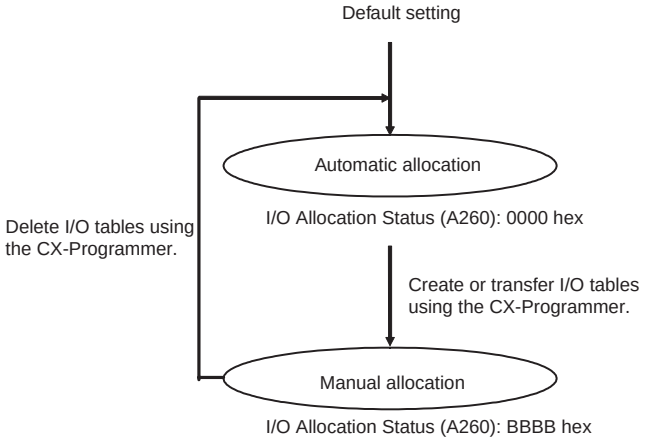
● Checking the I/O Allocation Method

You can use the I/O Allocation Status Word (A260) to check whether the I/O allocations have been set manually or automatically.

Address	Name	Contents	
A260	I/O Allocation Status	0000 hex	Automatic I/O allocation
		BBBB hex	Manual allocation

I/O Allocation Status Transitions

The I/O allocation method will change in the following manner.



When I/O tables are deleted using the CX-Programmer, the first word for each rack will be cleared at the same time that the I/O allocation status returns to automatic allocation. Also, the CPU Unit's System Setup Area will be initialized.

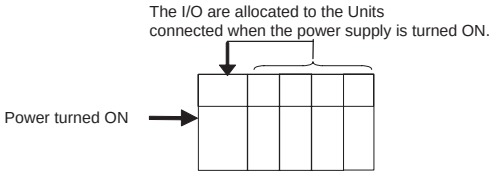
8-1-2 Automatic Allocation

Automatic Allocation (Default)

Use this method to allocate I/O according to the status of the connected Units. I/O are allocated automatically, so operations from a Programming Device is not required.

When automatic allocation is used, the I/O allocations will be updated every time the power supply to the PLC is turned ON.

To change from manual allocation to automatic allocation, delete the I/O tables using a Programming Device.



Additional information

Unlike with manual allocation, I/O tables are not created with automatic allocation, and so the connected Units are not verified.

Allocation Method

With automatic allocations, I/O will be allocated for the connected Units as described in the following table. No user operation is required.

Unit classification	Allocation method
Basic I/O Units	Allocated in order according to location.
Special I/O Units	Allocated according to front-panel rotary switch settings (i.e., unit numbers).
CPU Bus Units	

I/O Allocation Method for Each Unit Classification

This section describes the I/O allocation method for each Unit classification (Basic I/O Units, Special I/O Units, and CPU Bus Units).

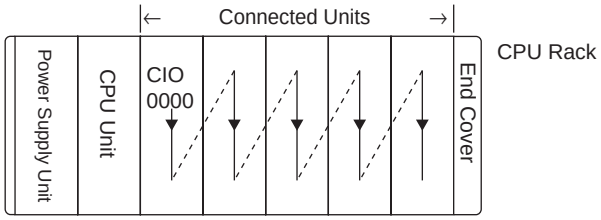
● I/O Allocation to Basic I/O Units

CJ-series Basic I/O Units are allocated words in the I/O Area (CIO 0000 to CIO 0159). They can be connected to the CPU Rack or Expansion Racks. The following method is used.

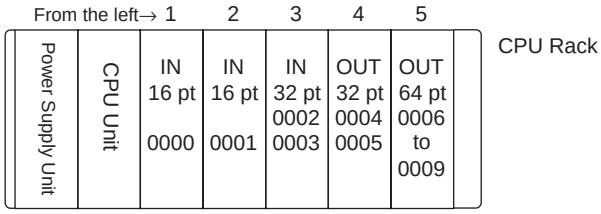
Basic I/O Units in the CPU Rack

Basic I/O Units on the CPU Rack are allocated words from left to right starting with the Unit closest to the CPU Unit. Each Unit is allocated as many words as it requires.

Note Units that have 1 to 16 I/O points are allocated16 bits and Units that have 17 to 32 I/O points are allocated 32 bits.



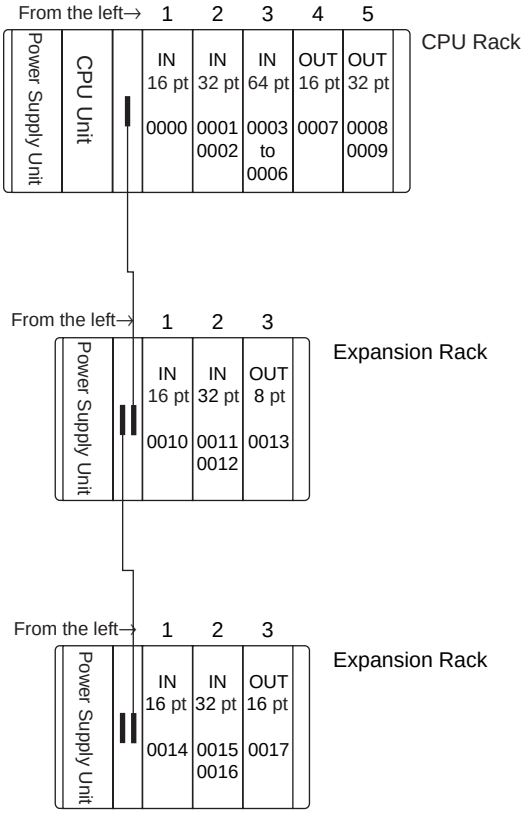
Example



Basic I/O Units in Expansion Racks

I/O allocation to Basic I/O Units continues from the CPU Rack to the Expansion Rack connected directly to the CPU Rack. Words are allocated from left to right and each Unit is allocated as many words as it requires, just like Units in the CPU Rack.

Example



● I/O Allocation to Special I/O Units

Each CJ-series Special I/O Unit is allocated ten words in the Special I/O Unit Area (CIO 2000 to CIO 2959) according the unit number set on the Unit. Special I/O Units can be mounted to the CPU Rack or Expansion Racks. The following table shows which words in the Special I/O Unit Area are allocated to each Unit.

Unit number	Number of words	Words allocated
0	10	CIO 2000 to CIO 2009
1	10	CIO 2010 to CIO 2019
2	10	CIO 2020 to CIO 2029
:	:	:
15	10	CIO 2150 to CIO 2159
: :	: :	: :
95	10	CIO 2950 to CIO 2959

Special I/O Units are ignored during I/O allocation to Basic I/O Units. Positions containing Special I/O Units are not allocated any words in the I/O Area.

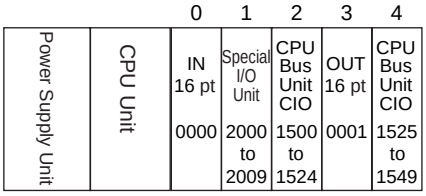
I/O Allocation to CPU Bus Units

Each CJ-series CPU Bus Unit is allocated 25 words in the CPU Bus Unit Area (CIO 1500 to CIO 1899) according the unit number set on the Unit. CPU Bus Units can be mounted to the CPU Rack or Expansion Racks. The following table shows which words in the CPU Bus Unit Area are allocated to each Unit.

Unit number	Number of words	Words allocated
0	25	CIO 1500 to CIO 1524
1	25	CIO 1525 to CIO 1549
2	25	CIO 1550 to CIO 1574
:	:	:
15	25	CIO 1875 to CIO 1899

CPU Bus Units are ignored during I/O allocation to Basic I/O Units. Positions containing CPU Bus Units are not allocated any words in the I/O Area. The unit numbers of CPU Bus Units are different from the unit numbers of Special I/O Units. Using the same unit number for both a CPU Bus Unit and a Special I/O Unit will not result in duplicate use of unit numbers.

Example



Slot	Unit	Model	Words required	Words allocated	Unit number	Group
0	16-point DC Input Unit	CJ1W-ID211	1	CIO 0000	---	Basic I/O Unit
1	Analog Input Unit	CJ1W-AD081	10	CIO 2000 to CIO 2009	0	Special I/O Unit
2	Serial Communica-tions Unit	CJ1W-SCU41	25	CIO 1525 to CIO 1549	1	CPU Bus Unit
3	16-point Transistor Output Unit	CJ1W-OD211	1	CIO 0001	---	Basic I/O Unit
4	Controller Link Unit	CJ1W-CLK21	25	CIO 1550 to CIO 1574	2	CPU Bus Unit
---	Built-in EtherNet/IP port on CPU Unit *1	CJ2H-CPU68-E1P	25	CIO 1500 to CIO 1524	0	CPU Unit (The port is treated as a CPU Bus Unit.)

*1 With the CJ2H-CPU6□-EIP and CJ2M-CPU3□, words are allocated to the built-in EtherNet/IP port as a CPU Bus Unit. These words are used to store the network communications status of the port. In the same way as with other CPU Bus Units, the words are allocated according to the unit number set with the rotary switches on the front of the CJ2H-CPU6□-EIP or CJ2M-CPU3□.

8-1-3 Manual Allocation

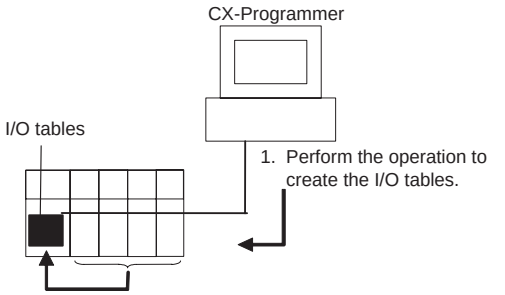
Manual Allocation

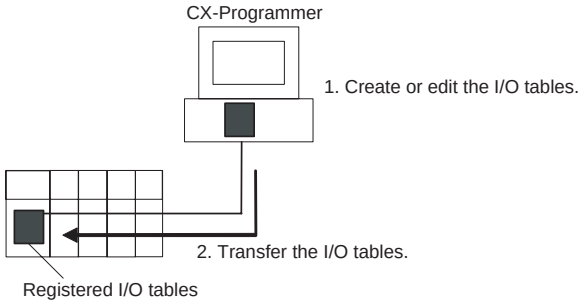
Use manual allocation to allocate I/O based on user-created I/O tables. If I/O tables are created, the connected Units will be verified against the I/O tables when the power supply is turned ON.

● Creating I/O Tables

Use one of the following methods to create the I/O tables.

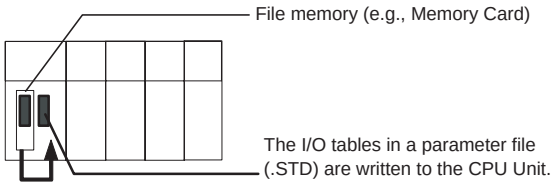
Creating I/O tables with Units connected	Creating I/O tables with User-set Allocations
Create the I/O tables with the Units connected and the CX-Programmer connected online.	Create the I/O tables as required using the CX-Programmer with the CX-Programmer offline, and transfer them to the CPU Unit.





Additional information

The PLC can be set to automatically transfer I/O tables saved in a Memory Card to the CPU Unit when the power supply is turned ON.



● Unit Verification

If I/O tables are created, the registered I/O tables are compared with the actual I/O at startup. If they do not match, A401.10 will turn ON to indicate an I/O setting error and operation will not be possible.

Creating I/O Tables with Units Connected

Allocation Method

I/O is allocated in the same way as for automatic allocation. Refer to *8-1-2 Automatic Allocation* for information on the allocation method.

I/O Allocation Procedure

With the CX-Programmer online, use the following procedure to create I/O tables for the connected Units.

1. Double-click ***IO Table*** in the project tree in the main window. The I/O Table Window will be displayed.
2. Select ***Options*** and then ***Create***. The models and positions of the Units mounted to the Racks will be written to the CPU Unit as the registered I/O tables.

Procedure for Comparing I/O Tables

Use the following procedure to compare the I/O tables with the physically connected Units while the CX-Programmer is online.

1. Double-click ***IO Table*** in the project tree in the main window. The I/O Table Window will be displayed.
2. Select ***Options*** and then ***Verify***. The models and positions of Units mounted to the Racks will be compared with the registered I/O tables.
3. Check the results when they are displayed.

Creating I/O Tables with User-specified Allocations

Allocation Method

Rather than allocating I/O automatically for the connected Units, I/O can be allocated manually using the following methods to create the desired I/O tables.

- Allocating words to Units at any positions on the Racks
- Reserving words
- Specifying first words on Racks
- Allocating I/O without the physical Units

Words are allocated as follows:

Unit classification	Allocation method
Basic I/O Units	There are some restrictions on settings, but the following are possible: The word that is allocated to any Unit on any Rack can be specified. Consecutive words will be allocated to following Unit in the order they are connected. This can be done to create 63 user-set allocation groups.
Special I/O Units	Words are allocated according to front-panel rotary switch settings (i.e., unit numbers).
CPU Bus Units	
Pulse I/O Blocks	The Pulse I/O Block closest to the CPU Unit is Pulse I/O Block 0 and the other one is Pulse I/O Block 1.

I/O Allocation Procedure

Create the I/O tables using the CX-Programmer, and then transfer them to the PLC. Create the I/O tables offline and then transfer them to the PLC online.

1. Double-click ***IO Table*** in the project tree in the main window. The I/O Table Window will be displayed.
2. Double-click the Rack to be edited. The slots for the selected Rack will be displayed.
3. Right-click the slot to which a Unit is to be assigned and select the Unit from the pull-down menu.
4. After editing the I/O tables, transfer them to the CPU Unit by selecting ***Options - Transfer to PLC***.

Allocating First Words to Rack Positions and Reserving Unused Words

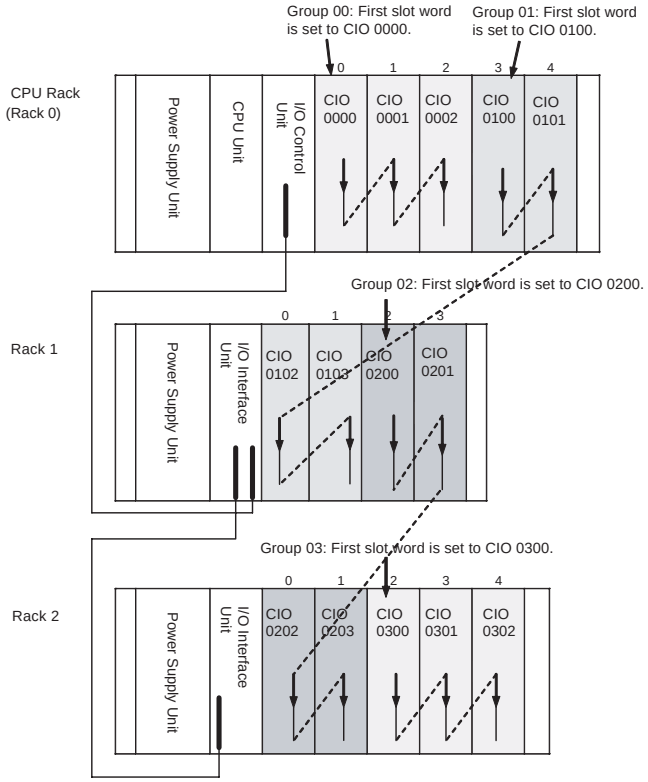
You can use the CX-Programmer to set the first word for a Unit at any position in any Rack. Thereafter, consecutive words after the specified first word will be allocated to the next Unit in the order the Units are connected. It is possible to make up to 63 groups by setting the first word for the first Unit in each group.

Allocation Method

When setting the first words for Rack positions (called “slots”), the first word must be set for slot 00 on the CPU Rack. The first word can then be set for any slot on any Rack for up to 63 other slots.

- Each first word set for a slot creates a group starting with that slot.
- Words are allocated starting from the specified word to the first Unit in the group and continuing left to right allocating consecutive words to each Unit until the next group (i.e., until the next Unit for which a first slot word is set). The next group can start on the same Rack or on a following Rack.
- For group 00, the first word is set for slot 00 of the CPU Rack. For groups 01 to 63, you can set the first word for any slot on any Rack.

For example, a first slot word has been set in the middle of each Rack. Only 16-point Units have been used.



First Slot Word Settings

Group	Rack	Slot	Word
00 *	CPU Rack	00	CIO 0000
01	CPU Rack	03	CIO 0100
02	Rack 1	02	CIO 0200
03	Rack 2	02	CIO 0300

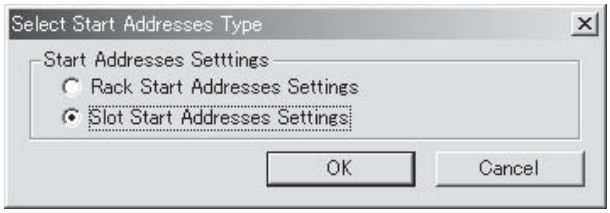
* Group 00 must start at slot 00 on the CPU Rack. Any word can be set. Any slot can be set on any Rack for groups 01 to 63.

Setting First Slot Words from the CX-Programmer

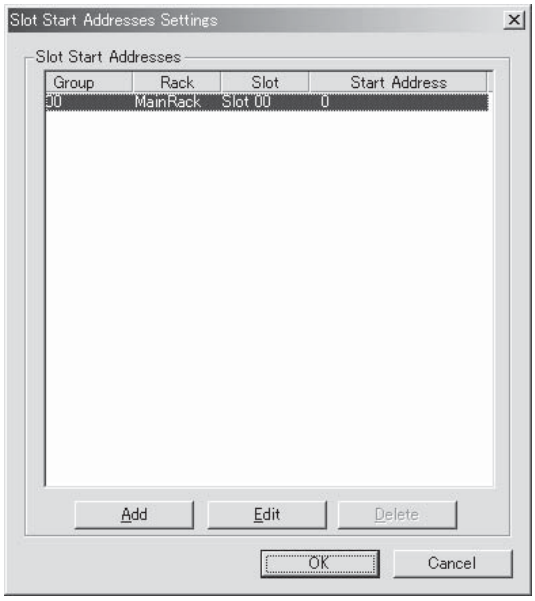
Use the following procedure to set the first rack words.

1 Select the ***Rack/Slot Start Addresses*** from the Option Menu on the I/O Table Window. The following dialog box will be displayed.

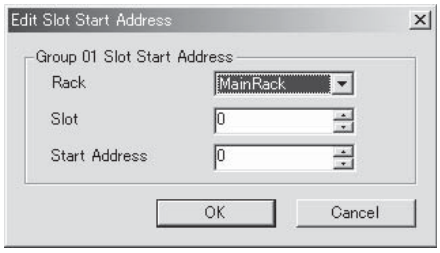
2 Select the *Slot Start Addresses Settings* Option and click the **OK** Button.



3 In the dialog box that will appear, set the first word for slot 00 on the CPU Rack.



4 To change the setting from CIO 0000, click the **Edit** Button. The follow dialog box will appear.



5 Set the desired word and click the **OK** Button.

6 To set slot first words for other groups, click the **Add** Button and make the appropriate settings for the Rack, slot, and word.

Setting	Setting range	Default	Remarks
Group	00 to 63	00	Groups numbers are allocated automatically in the order the groups are displayed and set.
Rack	CPU Rack (“Main Rack”) Racks 1 to 7	CPU Rack	Group 00 always starts at slot 00 on the CPU Rack.
Slot	00 to 03	0	
First word	0 to 999	0	---

Allocating First Words to Racks

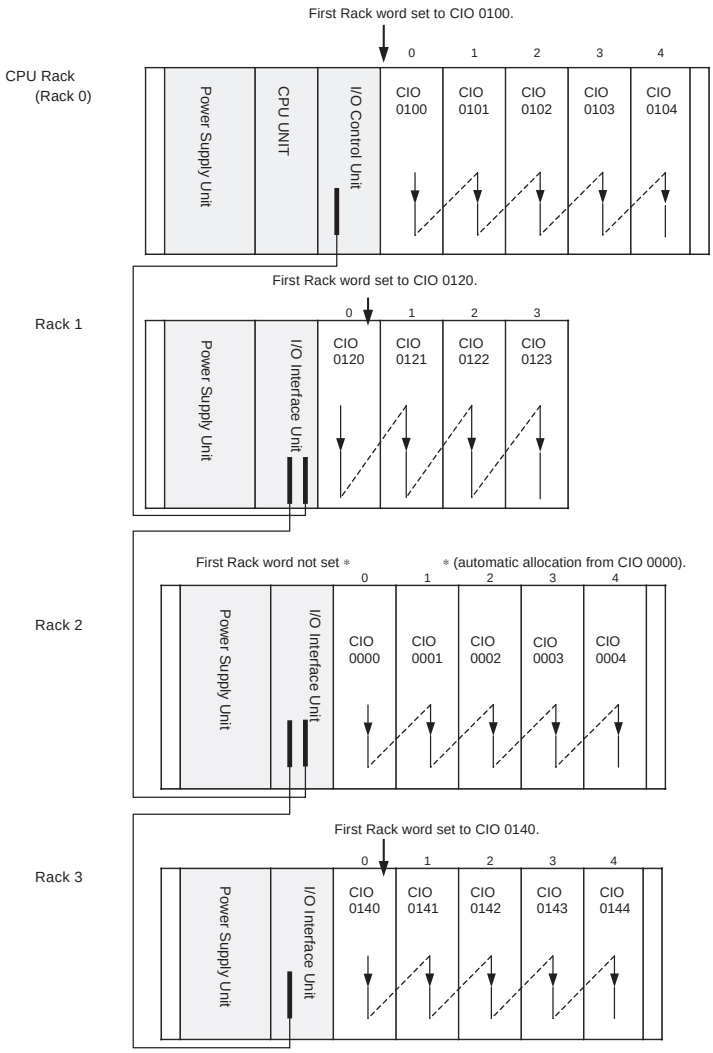
In the CJ-series PLCs, the first word allocated to each Rack can be set with the CX-Programmer's I/O table edit operation. For example, the CPU Rack can be set to be allocated words starting with CIO 0000; the next Rack, words starting with CIO 0100; the next Rack, words starting with CIO 0200; etc. This can make it easier to check word allocations to Units without calculating all the way from the CPU Rack.

● Word Allocations

For Racks in which the first word address has been set, words are allocated to Units in the order that the Units are mounted (from left to right) beginning with the specified first word. Words are not allocated to empty slots.

For Racks in which the first word address has not been set, words are allocated in rack-number order (lowest to highest) continuing from the last word allocated to the previous rack and starting with CIO 0000 on the first Rack for which the first word is not set.

Example: Setting the First Words for Racks



Rack First Word Settings

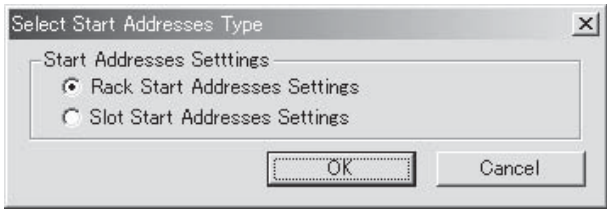
Rack	First word
CPU Rack	CIO 0100
Rack 1	CIO 0120
Rack 2	0000
Rack 3	0140

Note Rack numbers (0 to 3) are fixed according to the order that the Racks are physically connected with cable. The CPU Rack is always Rack 0 and the other Racks are, in order, Racks 1 to 3. These numbers cannot be changed.

Setting First Rack Words from the CX-Programmer

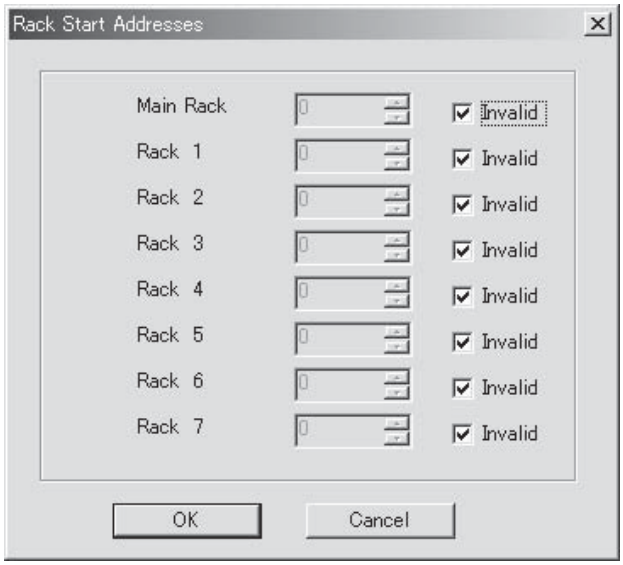
Use the following procedure to set the first rack words.

1 Select the ***Rack/Slot Start Addresses*** from the Option Menu on the I/O Table Window. The following dialog box will be displayed.



2 Select the Rack Start Addresses Settings Option and click the **OK** Button.

3 In the dialog box that will appear, remove the check marks from the settings disabling the first rack word settings and set the address of the first words for the CPU Rack and Expansion Racks (1 to 7).



Setting	Setting range	Default	Remarks
Rack Start Address	0 to 900	0	Same for all Racks
Invalid	Selected or cleared	Selected (invalid)	

Note 1 Up to 3 Racks can be set for any CPU Unit model.

2 Although the CX-Programmer window will display 7 Racks, only 3 Racks can be set for the CJ2 CPU Unit.

Precautions for Correct Use

- Be sure to make the first word settings so that allocated words do not overlap. The first word setting for a Rack can be any address from CIO 0000 to CIO 0900. If the same word is allocated to two Racks or if the first word set for the two Racks is CIO 0901 or higher, the I/O tables cannot be created and the Expansion Rack Number Duplication Flags (A409.00 to A409.03: Rack numbers 0 to 3) will turn ON. The Duplication Error Flag (fatal error) (A401.13) will also turn ON.
- Always register the I/O tables after setting the first word allocation for a Rack. To register the I/O tables, select ***Options*** and then ***Create*** in the I/O Table Window of the CX-Programmer. The I/O Table Registration operation registers the I/O words allocated to the Racks.
- If the actual system configuration is changed after registering the I/O tables so that the number of words or Unit models does not match those in the I/O tables, an I/O Verification Error (A402.09) or I/O Setting Error (A401.10) will occur. A CPU Bus Unit Setting Error (A402.03) or Special I/O Unit Setting Error (A402.02) may occur as well. If a Unit is removed, words can be reserved for the missing Unit using the I/O Table Change Operation. If a Unit is changed or added, all of the words in the program following that Unit's allocated words will be changed and the I/O Table Registration Operation will have to be performed again.
- If the I/O tables are deleted using the CX-Programmer, the first word settings for the Racks will also be deleted.

8-1-4 I/O Table Errors and Precautions

I/O Table Errors

● I/O Setting Errors and Number Duplication Errors

I/O tables cannot be created in the following cases when the procedures to create, edit, or transfer I/O tables (i.e., using user-set I/O allocations) have been performed.

Problem	Error	Description	Correction
Verification error between connected Units and I/O tables	I/O SET ERR (fatal error)	The I/O Setting Error Flag (fatal error) (A401.10) will turn ON if there is a verification error, that is, if the registered I/O tables do not match the models and positions of the Units actually connected in the basic system (i.e., CPU Racks and Expansion Racks). Operation will not be possible. The RUN indicator on the front of the CPU Unit will turn OFF, and the ERR/ALARM indicator will flash red.	• If the number of Units is not correct, turn OFF the power supply and correctly connect the proper Units. • If the number of Units is correct, confirm the Unit discrepancy by comparing I/O tables with the CX-Programmer, turn OFF the power supply, and then correct the Unit connections. • If there is a mistake in the I/O tables, recreate or edit them using the CX-Programmer to correct the mistake.
The same unit number for Special I/O Units or CPU Bus Units has been set twice.	Duplication Error (fatal error)	The Duplication Error Flag (A401.13) will turn ON in any of the following cases. • The same unit number is set for more than one CPU Bus Unit. • The same unit number is set for more than one Special I/O Unit • The same word is allocated to more than one Basic I/O Unit. • The same rack number is set for more than one Expansion Rack. Operation will not be possible. The RUN indicator on the front of the CPU Unit will turn OFF, and the ERR/ALARM indicator will flash red.	Check the unit numbers of the CPU Bus Units or Special I/O Units, eliminate the duplications, and turn the Rack's power supply OFF and then ON again. Check allocations to Units on the rack number whose bit in ON in A409.00 to A409.03. Correct the allocations so that no words are allocated more than once, including to Units on other Racks, and turn the Rack's power supply OFF and then ON again. Check the first word setting for the Rack indicated in A409.00 to A409.03 and change the setting to a valid word address below CIO 0900 with the CX-Programmer.

In addition, I/O tables cannot be used when the Too Many I/O Points Flag (fatal error) (A401.11) is ON or the I/O Bus Error Flag (fatal error) (A401.14).

● Detailed Information on I/O Table Creation Errors

The contents of A261 (I/O Table Creation Error Details) provides information on the Unit causing the error whenever one occurs when creating the I/O tables from the CX-Programmer. This information will make it easier to find the Unit causing the problem with troubleshooting I/O tables.

Name		Address		Contents
		Word	Bit	
Detailed Information on I/O Table Creation Errors	CPU Bus Unit Setup Area Initialization Error Flag	A261	00	ON: Error in CPU Bus Unit Setup Turns OFF when I/O tables are generated normally.
	I/O Overflow Flag		02	ON: Overflow in maximum number of I/O points. Turns OFF when I/O tables are generated normally.
	Duplication Error Flag		03	ON: The same unit number was used more than once. Turns OFF when I/O tables are generated normally.
	I/O Bus Error Flag		04	ON: I/O bus error Turns OFF when I/O tables are generated normally.
	Special I/O Unit Error Flag		07	ON: Error in a Special I/O Unit Turns OFF when I/O tables are generated normally.
	I/O Unconfirmed Error Flag		09	ON: I/O detection has not been completed. Turns OFF when I/O tables are generated normally.

Precautions in Setting I/O Tables

After setting I/O tables, check for any duplications in word allocations if Units are to be changed. It is conceivable, however, that duplications in word allocations could occur after the I/O tables have been registered, e.g., as the result of replacing a 1-word Unit with a 2-word Unit. In this case, the extra word needed by the new Unit would still also be allocated to the next Unit.

When the PLC is turned ON after Units have been changed, the CPU Unit checks the registered I/O tables against the actual Units connected in the PLC. If there are any duplications, an error will occur and it will be no longer possible to edit the I/O tables. If this happens, it will be necessary to edit the I/O tables and transfer them again.

When I/O tables are edited, the CX-Programmer checks for any duplication in first word settings and setting range errors to help eliminate problems.

Precautions when Using a Memory Card

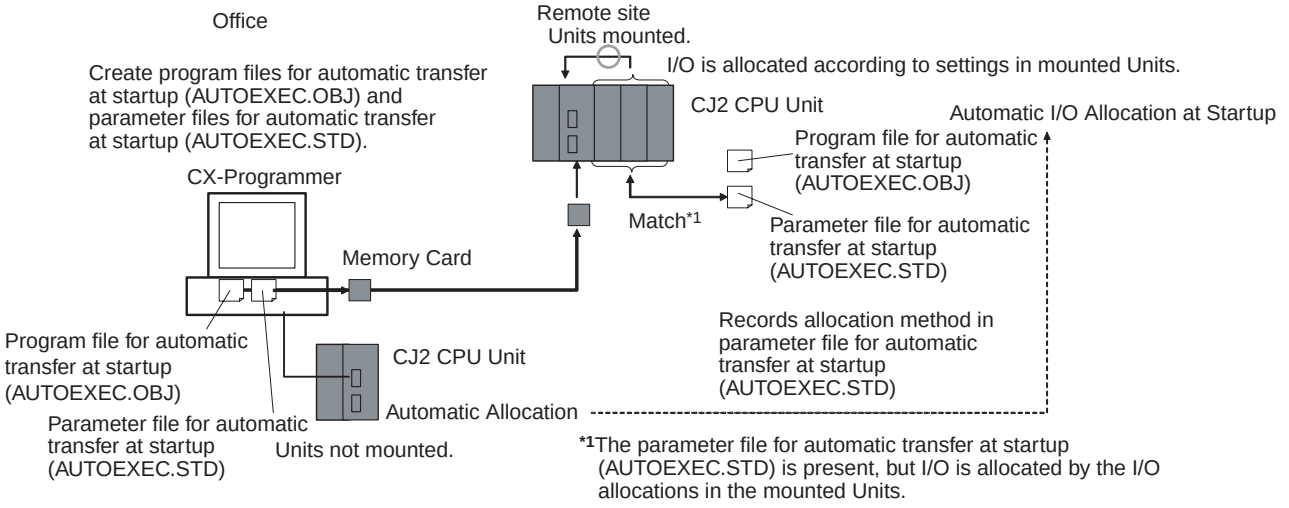
The I/O allocation method used to create the CPU Unit's I/O table (automatic I/O allocation at startup or user-set I/O allocation) is recorded in the parameter file for automatic transfers at startup (AUTOEXEC.STD). When automatic transfer at startup is executed from the Memory Card, the recorded method is automatically detected and used to allocate I/O automatically at startup or verify the I/O table.

The descriptions below explain the two different methods used to create the I/O tables in the CPU Unit by creating a parameter file for automatic transfer at power startup (AUTOEXEC.STD).

- Automatic Allocation*1
  The I/O tables in the parameter file for automatic transfer at startup in the Memory Card are disabled and I/O is allocated using automatic I/O allocation at startup based on the Units actually connected in the PLC.

*1 With automatic I/O allocation at startup, I/O tables are not created in advance and I/O allocations are automatically made to the Basic I/O Units that are actually connected each time the power supply is turned ON.

- Manual Allocation
  The I/O tables in the parameter file for automatic transfer at startup in the Memory Card are enabled, and the CPU Unit verifies the I/O tables against the Units actually connected in the PLC.



8-2 Setting CPU Bus Units and Special I/O Units

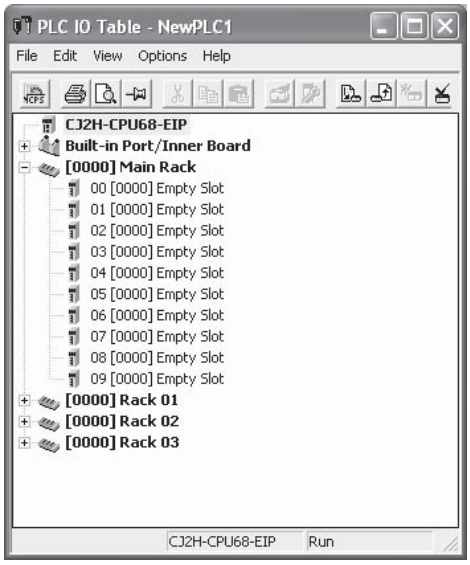
8-2-1 Setting Parameters

The CX-Programmer can be used to set parameters in the DM Area and CIO Area words allocated to CJ-series CPU Bus Units and Special I/O Units and to set the CPU Bus Unit Setup for EtherNet Units. The I/O Table Window is used in the CX-Programmer.

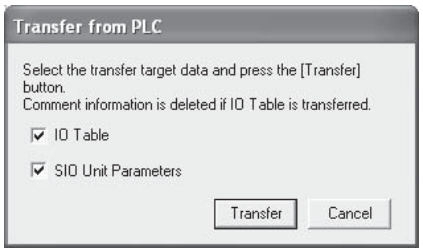
The settings can be made either online or offline. If they are made offline, you must go online to transfer them.

This section describes an example of editing the I/O tables and making settings for CPU Bus Units and Special I/O Units after creating the I/O tables and transferring them from the CPU Unit to the I/O Table Window. The procedure is the same as when registering CPU Bus Units and Special I/O Units to the I/O tables offline and then editing the settings for CPU Bus Units and Special I/O Units.

1 Connect the CX-Programmer online and open the I/O Table Window.

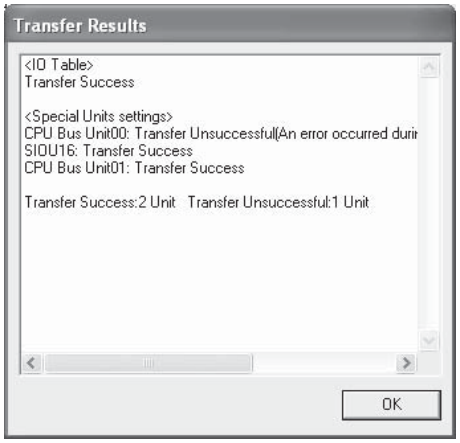


2 Select *Options – Transfer from the PLC* from the I/O Table Window menu. The Transfer from the PLC Dialog Box will be displayed as shown in the following figure.

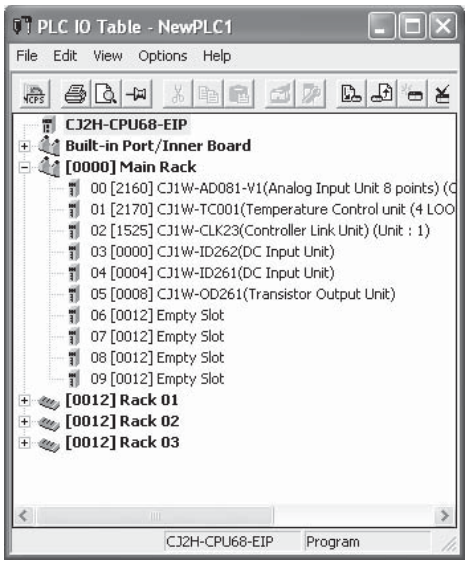


At this point, the *I/O Table* and *SIO Unit Parameters* Options can be selected.

3 Select the options (check the boxes) for the data that you want to transfer and click the Transfer Button. In this case, both options have been selected. The I/O table and Unit parameters data will be transferred from the PLC and the transfer results will be displayed.

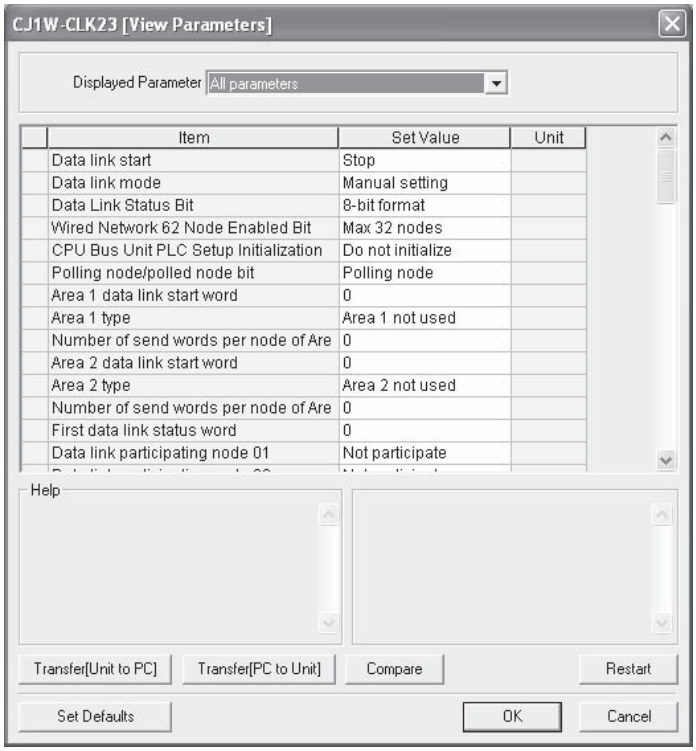


4 Check the transfer results and click the OK Button. The I/O table transferred from the PLC will be displayed.

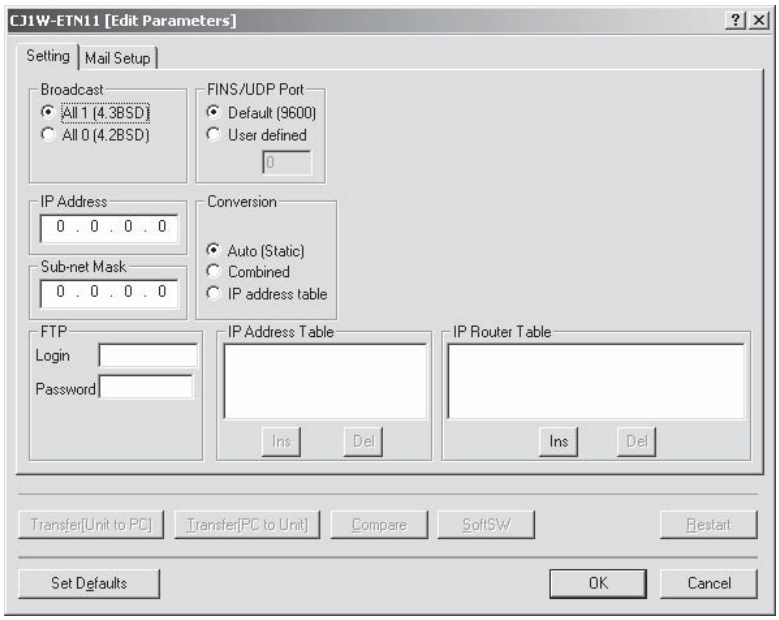


5 To edit the parameters of a Special I/O Unit or CPU Bus Unit, select the Unit and either double-click the Unit or select *Edit* – *SIO Unit Parameters* from the menu. The selected Unit's Edit Parameters Dialog Box will be displayed.

- In this case, a Controller Link Unit's Edit Parameters Dialog Box has been displayed to edit the Unit's CPU Bus Unit Allocation DM Settings.

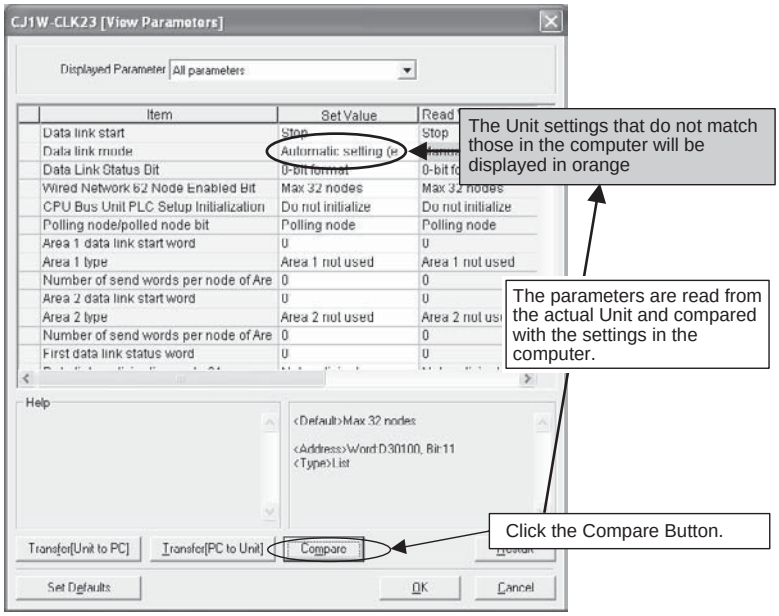


- In this case, an EtherNet Unit's Edit Parameters Dialog Box has been displayed to edit the CPU Bus Unit Settings in the words allocated in the DM Area.



6 Edit the parameters and, if necessary, click the **Compare** Button. If the **Compare** Button is clicked while the PLC is online, the software immediately checks whether the I/O table settings match the Unit settings (e.g., in the words allocated in the DM Area and CIO Area) in the CPU Unit. If the contents do not match, the settings read from the actual Unit will be displayed in orange.

The following example shows the results of comparing settings for a Controller Link Unit.



7 If you want to download the parameters set for each Unit to the CPU Unit, click the **Download to Unit** Button.

Precautions for Correct Use

When Special I/O Unit or CPU Bus Unit settings are edited in the I/O Table Window, the parameters allocated in the DM Area or CPU Bus Unit Setup Area for the Unit are only transferred to the actual PLC when the *Transfer PC to Unit* Button is clicked in the Edit Parameters Dialog Box. They are not automatically enabled and will not be used unless you enable them. Click the *Reset* Button to enable the parameter settings that have been transferred.

Additional information

Storing the Parameter Settings Data File

- The data set in the Edit Parameters Dialog Box can be saved in a parameter settings file with an xml filename extension. To save the parameter settings, right-click the desired Special I/O Unit or CPU Bus Unit in the I/O table and select *Save Parameters* from the pop-up menu.
- The saved parameter settings can also be read from a parameter settings file (xml filename extension), but the settings must be for the same model Special I/O Unit or CPU Bus Unit.

8-2-2 Data Exchange

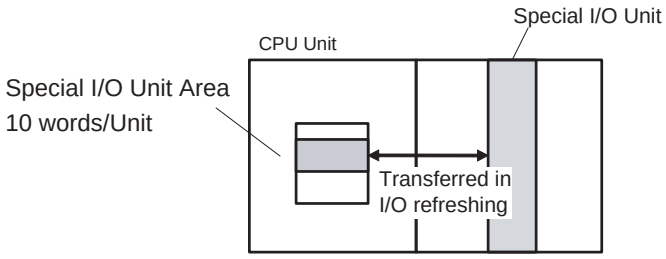
This section describes how data can be exchanged between Special I/O Units or CPU Bus Units, and the CPU Unit.

Special I/O Units

Special I/O Unit Area (I/O Refreshing)

Data is exchanged each cycle during I/O refreshing of the Special I/O Unit Area. Basically, 10 words are allocated to each Special I/O Unit based on its unit number setting. Refer to the operation manuals for individual Special I/O Units for details.

The Special I/O Unit Area ranges from CIO 2000 to CIO 2959 (10 words × 96 Units).



Transfer of Words Allocated in DM Area

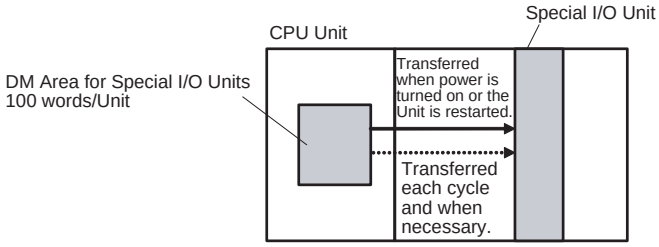
There are three times that data may be transferred through the words allocated to each Unit. The timing of data transfers depends on the model being used.

- Data transferred when the PLC is turned ON.
- Data transferred when the Unit is restarted.
- Data transferred when necessary.

Some models transfer data in both directions, from the DM Area to the Unit and from the Unit to the DM Area. See the Unit's Operation Manual for details on data transfers.

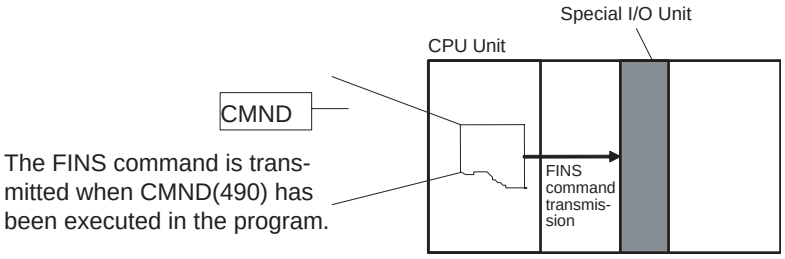
Special I/O Unit Words in the DM Area: D20000 to D29599 (100 Words × 96 Units)

Data is transferred to these words in the DM Area for initial settings for Special I/O Units. When the contents of this allocated words are changed from the program to change the system, the Special I/O Unit Restart Bit (A502.00 to A507.15) must be turned ON to restart the Unit.

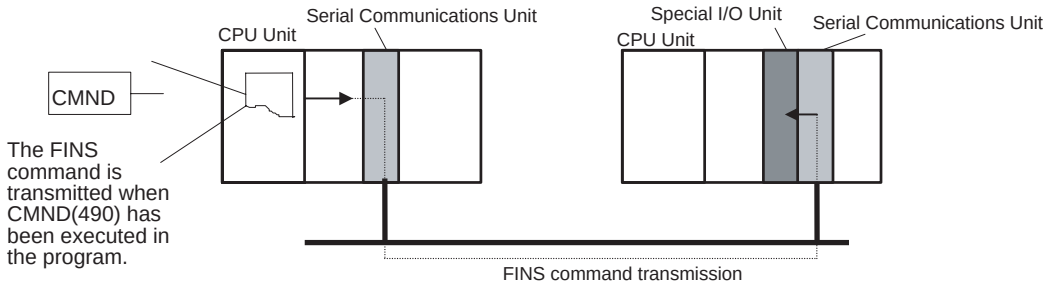


FINS Commands

The CMND(490) instruction can be added to the ladder program to send a FINS command to the Special I/O Unit.



Note FINS commands can be transmitted to Special I/O Units in other PLCs in the network, not just the local PLC.



Special I/O Unit Initialization

Special I/O Units are initialized when the PLC's power is turned ON or the Unit's Restart Bit (A502.00 to A507.15) is turned ON. The Unit's Special I/O Unit Initialization Flag (A330.00 to A335.15) will be ON while the Unit is initializing.

I/O refreshing (cyclic I/O refreshing or refreshing by IORF(097) or FIORF(225) will not be performed for a Special I/O Unit while its Initialization Flag is ON.

Disabling Special I/O Unit Cyclic Refreshing

Ten words are allocated to each Special I/O Unit in the Special I/O Unit Area (CIO 2000 to CIO 2959) based on the unit number set on the front of each Unit. The data in the Special I/O Unit Area is refreshed in the CPU Unit every cycle during I/O refreshing (just after execution of the END(001) instruction).

I/O refreshing may take too long if too many Special I/O Units are installed. If I/O refreshing is taking too much time, the PLC Setup can be set to disable cyclic refreshing for particular Special I/O Units. (The Special I/O Unit Cyclic Refreshing Disable Bits are in PLC Setup addresses 226 to 231.)

If the I/O refreshing time is too short, the Unit's internal processing will not be able to keep pace, the Special I/O Unit Error Flag (A402.06) will be turned ON, and the Special I/O Unit may not operate properly. In this case, the cycle time can be extended by setting a minimum cycle time in the PLC Setup or cyclic I/O refreshing with the Special I/O Unit can be disabled.

Then cyclic refreshing has been disabled, the Special I/O Unit's data can be refreshed during program execution with IORF(097) or FIORF(225).

Precautions for Correct Use

IORF(097), FIORF(225), IORD (222), and IOWR(223) can be executed for Special I/O Units from interrupt tasks. When doing so, always disable the Special I/O Unit's cyclic refreshing in the PLC Setup. If cyclic refreshing is not disabled and either of the following processes is executed in an interrupt task, a non-fatal error will occur and the Duplicate Refresh Error Flag (A402.13) will turn ON.

- I/O refreshing is executed using IORF(097) or FIORF(225) for the same Special I/O Unit.
- Data is read or written to or from the memory area using IORD (222) or IOWR(223) for the same Special I/O Unit.

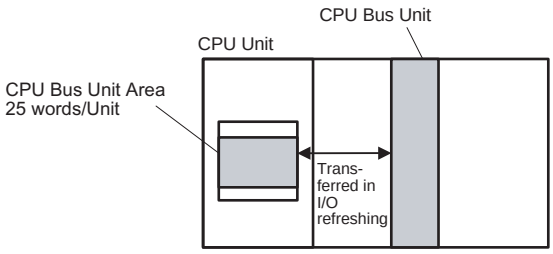
Whenever disabling a Special I/O Unit's cyclic refreshing, be sure that the I/O for that Unit is refreshed with IORF(097) or FIORF(225) in the program at least every 11 seconds during operation. A CPU Unit service monitoring error will occur in the Special I/O Unit if it is not refreshed every 11 seconds.

CPU Bus Units

● CPU Bus Unit Area (I/O Refreshing)

Data is exchanged each cycle during I/O refreshing of the CPU Bus Unit Area. Basically, 25 words are allocated to each CPU Bus Unit based on its unit number setting. The number of words actually used by the CPU Bus Unit varies.

The Special I/O Unit Area ranges from CIO 1500 to CIO 1899 (25 words × 16 Units).



Note The CPU BUS I/O REFRESH instruction (DLNK(226)) can be executed in the ladder program to refresh the CIO Area words allocated to the CPU Bus Unit of a specified unit number.

● Transfer of Words Allocated in the DM Area

One hundred words are allocated to each Unit according to the unit number.

Note These words are not used for all models of CPU Bus Unit.

DM Area Words for the CPU Bus Units: D30000 to D31599 (100 words × 16 Units)

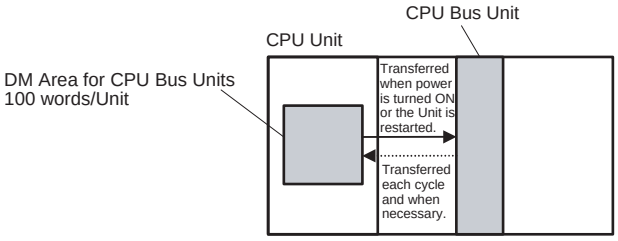
The contents of these words are transferred at the following times. Which transfers actually take place depends on the model of the Unit.

- Data transferred when the PLC is turned ON.
- Data transferred each cycle.
- Data transferred when necessary.

Note The CPU BUS I/O REFRESH instruction (DLNK(226)) can be executed in the ladder program to refresh the DM Area words allocated to the CPU Bus Unit of a specified unit number.

Some models transfer data in both directions, from the DM Area to the Unit and from the Unit to the DM Area. Refer to the Unit's operation manual for details on data transfers.

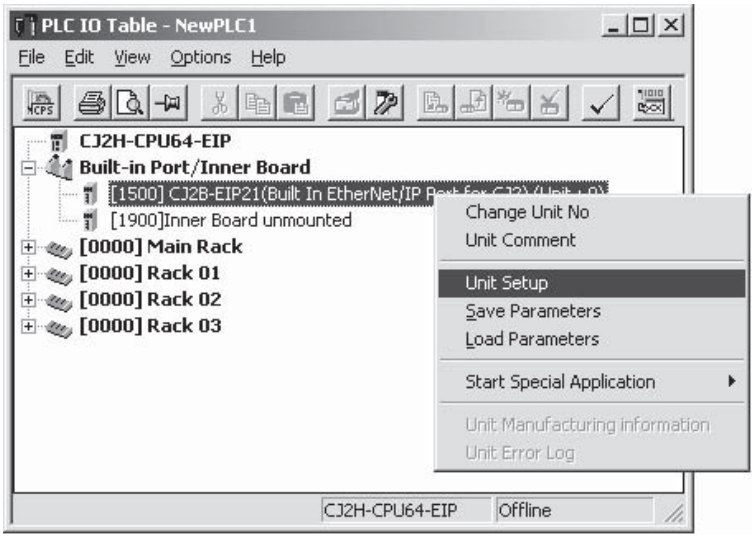
These 100 words are generally used to hold initial settings for the CPU Bus Unit. When the contents of this area are changed from the program to reflect a change in the system, the Restart Bits (A501.00 to A501.15) for affected Units must be turned ON to restart the Units.



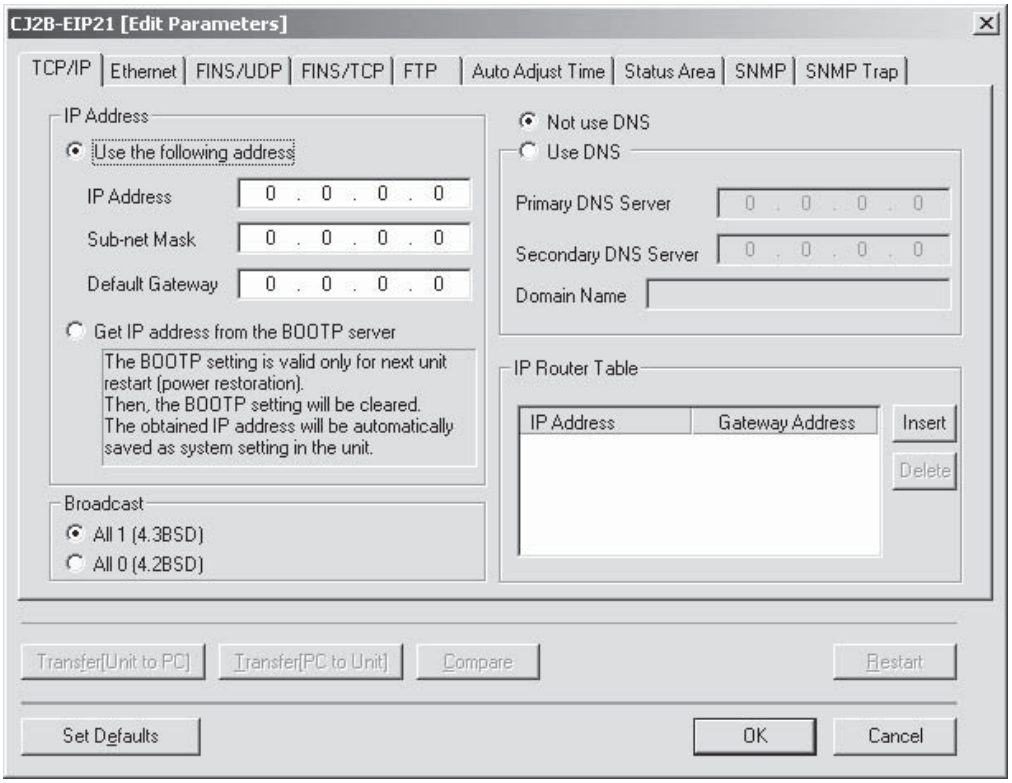
Additional information

The built-in EtherNet/IP port of the CJ2H-CPU6□-EIP or CJ2M-CPU3□ is treated as a CPU Bus Unit according to the rotary switch setting. Use the following procedure to set the communications parameters.

(1) Select CJ2B-EIP21 (built-in EtherNet/IP port for CJ2H) or CJ2M-EIP21 (built-in EtherNet/IP port for CJ2M) under Built-in Port/Inner Board in the I/O tables of the PLC. Right-click and select Unit Setup.



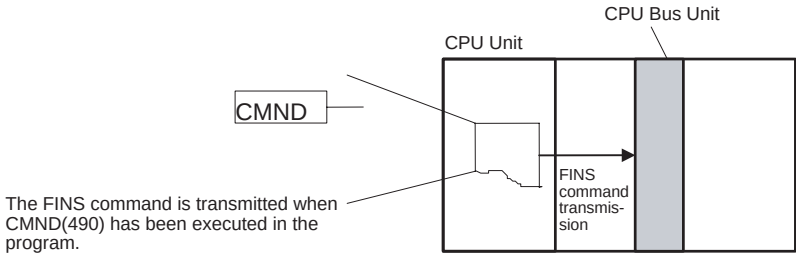
(2) Set the parameters.



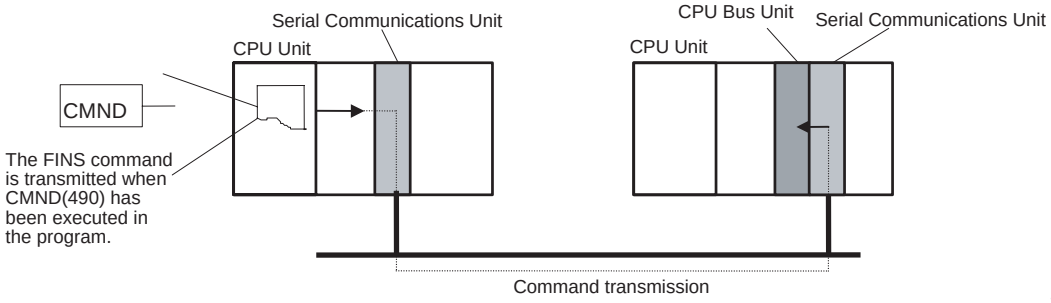
For information on parameters of the built-in EtherNet/IP port, refer to the *EtherNet/IP Units Operation Manual* (Cat. No. W465).

● FINS Commands

The CMND(490) instruction can be added to the ladder program to issue a FINS command to the CPU Bus Unit.



Note FINS commands can be transmitted to CPU Bus Units in other PLCs in the network, not just the local PLC.



● CPU Bus Unit Initialization

CPU Bus Units are initialized when the PLC's power is turned on or the Unit's Restart Bit (A501.00 to A501.15) is turned ON. The Unit's CPU Bus Unit Initialization Flag (A302.00 to A302.15) will be ON while the Unit is initializing.

Cyclic I/O refreshing will not be performed for a CPU Bus Unit while its Initialization Flag is ON.

9

PLC Setup

This section describes the parameters in the PLC Setup, which are used to set options for PLC operation.

9-1 Overview of the PLC Setup 9-2

9-2 PLC Setup Settings 9-4

9-3 PLC Setup Settings 9-5

9-3-1 Startup Operation Settings 9-5
9-3-2 CPU Unit Settings 9-8
9-3-3 Timings/Synchronous Settings 9-14
9-3-4 Special I/O Unit Cyclic Refreshing 9-19
9-3-5 Basic I/O Unit Rack Response Times 9-21
9-3-6 Serial Port Settings 9-23
9-3-7 Peripheral Service 9-30
9-3-8 FINS Protection 9-31

9-1 Overview of the PLC Setup

The PLC Setup contains basic CPU Unit software parameter settings that the user can change to customize PLC operation. These settings can be changed from a Programming Console or other Programming Device. Change the PLC Setup in the following cases.

Application	Parameter
The input response time settings for CJ-series Basic I/O Units must be changed in the following cases: • Chattering or noise occur in Basic I/O Units. • Short pulse inputs are being received for intervals longer than the cycle time.	Basic I/O Unit Rack Response Time
Data in all regions of I/O Memory (including the CIO Area, Work Areas, Timer Flags and PVs, Task Flags, Index Registers, and Data Registers) must be retained when the PLC's power is turned on.	IOM Hold Bit Status at Startup
The status of bits that are force-set or force-reset from a Programming Device must be retained when the PLC's power is turned ON.	Forced Status Hold Bit Status at Startup
• Debugging the PLC, • Changing the Startup Mode to PROGRAM or MONITOR mode.	Startup Mode
Detection of low-battery errors is not required when using battery-free operation.	Detect Low Battery
Disabling detection of duplicate refreshing.	Detect Duplicate Refresh Errors
The RS-232C port will not be used with the Programming Console or CX-Programmer (peripheral bus) communications speed auto-detection and will not use the default host link communications settings such as 9,600 bps.*1	RS-232C Port Settings
You want to speed up communications with a PT via an NT Link.	Set the peripheral port or the RS-232C port communications port baud rate to “high-speed NT Link.”
You want the intervals for scheduled interrupt function to be set in units of 1 ms (or 0.1 ms) rather than 10 ms.	Scheduled Interrupt Time Units
You want to use high-speed interrupt function for an I/O interrupt task or scheduled interrupt task.	Enable High-speed Interrupt Function
You want to use the synchronous unit operation function.	Use Synchronous Operation
Finding instruction errors when debugging.	Stop CPU on Instruction Error
You want a minimum cycle time setting to create a consistent I/O refresh cycle.	Minimum Cycle Time
You want to set a maximum cycle time other than 1 second (10 ms to 40,000 ms).	Watch Cycle Time
Performing special processing when power is interrupted.	Power OFF Interrupt Task*2
You want to delay the detection of a power interruption.	Power OFF Detection Delay Time*1
• You want to execute IORF or FIORF in an interrupt task. • You want to shorten the average cycle time when a lot of Special I/O Units are being used. • You want to extend the I/O refreshing interval for Special I/O Units.	Special I/O Unit Cyclic Refreshing

Application	Parameter
You do not want to record user-defined errors for FAL(006) and FPD(269) in the error log.	FAL Error Log Registration
You want to reduce fluctuation in the cycle time caused by text string processing	Background Execution for Table Data, Text String, and Data Shift Instructions
• There are Units that take time to start when the power supply is turned ON. • You do not want to wait for Units to complete startup processing to start CPU Unit operation.	Execution Setting

*1 Pin 5 of the DIP switch on the front of the CPU Unit must be OFF to change the PLC Setup settings.

*2 These settings cannot be used if the CJ1W-PD022 is mounted.

● Related Bits and Words in the Auxiliary Area

Name	Address	Description	Access
PLC Setup Error Flag (Non-fatal error)	A40210	ON when there is a setting error in the PLC Setup.	Read-only

9-2 PLC Setup Settings

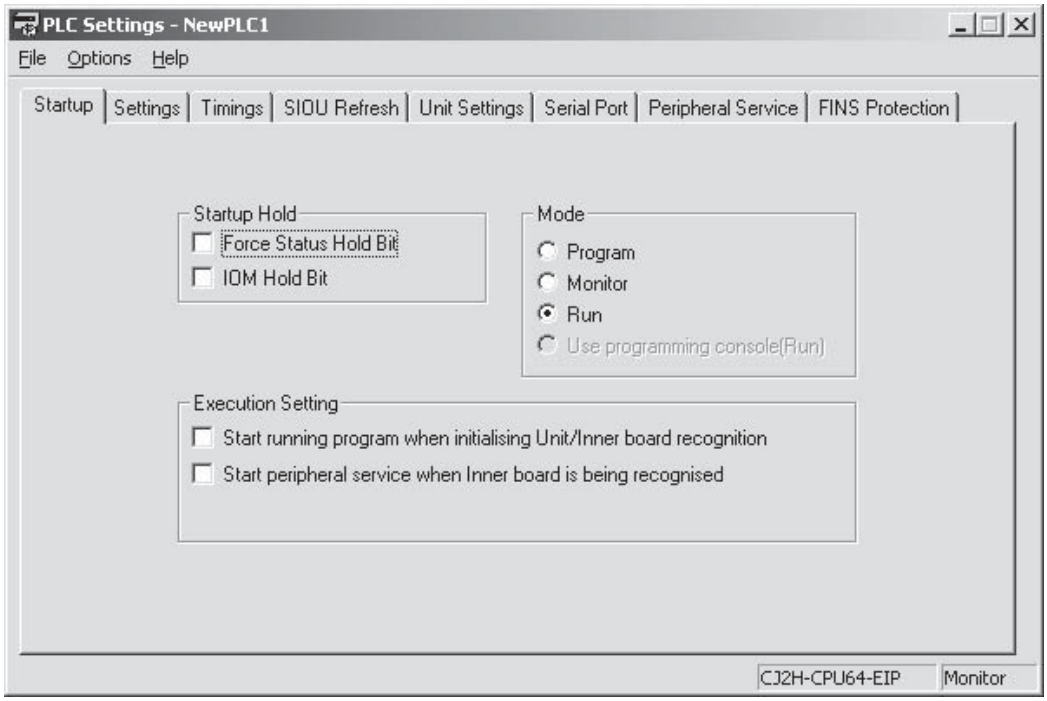
The following table gives the default settings in the PLC Setup. To change the settings, edit the PLC Setup with the CX-Programmer, and then transfer the PLC Setup to the CPU Unit.

CX-Programmer PLC Setup Tab Page	Setting name		Applicable CPU Units		Default	Page
			CJ2H CPU Units	CJ2M CPU Units		
Startup	Startup Hold Settings	Forced Status Hold Bit	Yes	Yes	Not retained when power is turned ON.	9-5
		IOM Hold Bit	Yes	Yes	Not retained when power is turned ON.	9-6
	Operating Mode		Yes	Yes	RUN mode	9-7
	Execution Setting	Start running program before initializing Unit/Inner board recognition	Yes	Yes	Do not start.	9-7
CPU Unit Settings	Execute Process Settings	Do not detect Low Battery	Yes	Yes	Detect.	9-8
		Detect Duplicated Refreshing Error	Yes	Yes	Detect.	9-10
		Stop CPU on Instruction Error	Yes	Yes	Do not stop.	9-10
		FAL Error Log Registration	Yes	Yes	Register to error log.	9-11
	Background Execution Settings		Yes	Yes	Not executed in background.	9-12
	Comms Instructions Settings in FB	Retry Counts	Yes	Yes	0 times	9-13
		Response Timeout	Yes	Yes	2 s	9-13
Timings/Synchronous Settings	Watch Cycle Time		Yes	Yes	1,000 ms (1 s)	9-14
	Constant Cycle Time (Minimum Cycle Time)		Yes	Yes	Not Constant	9-15
	Scheduled Interrupt Interval		Yes	Yes	10 ms	9-16
	Power Off Detection Time		Yes	Yes	0 ms	9-17
	Power Off Interrupt		Yes	Yes	Do not use.	9-18
	Enable High-speed Interrupt Function		No	Yes	Do not enable.	9-18
	Use Synchronous Operation		No	Yes	Do not use.	9-18
Special I/O Unit Cyclic Refreshing	Disable SIOU Cycle Refresh		Yes	Yes	Not disabled.	9-19
Unit Settings	Input response times for Basic I/O Units		Yes	Yes	8 ms	9-19
Serial Port	Mode (Pin 5 on the DIP switch on the CPU Unit must be OFF (default) to set the mode.)		Yes*	Yes	Host Link	9-23
Peripheral Service	Execution Mode		Yes	Yes	Normal	9-30
	Set Time to All Events		Yes	Yes	10% of cycle time	9-30
FINS Protection	Settings for FINS write protection via network		Yes	Yes	FINS write protection disabled.	9-31

* Cannot be selected for Serial PLC Links.

9-3 PLC Setup Settings

9-3-1 Startup Operation Settings



Startup Hold Settings

● Forced Status Hold Bit Startup Hold Setting

Use this parameter to set whether to retain the Forced Status Hold Bit (A500.13) in the Auxiliary Area at startup.

Parameter	Settings	Default	Function	Related flags and words
Forced Status Hold Bit Startup Hold Setting	OFF: Cleared ON: Retained	OFF	This setting determines whether the status of the Forced Status Hold Bit (A500.13) is retained at startup.	A500.13 (Forced Status Hold Bit)

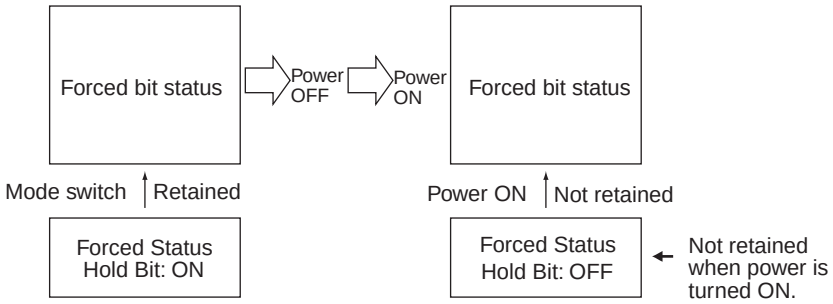
- Relation between Forced Status Hold Bit at Startup and this Parameter

The Forced Status Hold Bit (A500.13) can be turned ON to retain the forced status of all bits that have been force-set or force-reset when the CPU Unit's operating mode is switched between PROGRAM mode and MONITOR mode. When the PLC is turned ON, the Forced Status Hold Bit itself will be turned OFF unless it is protected with this PLC Setup parameter setting.

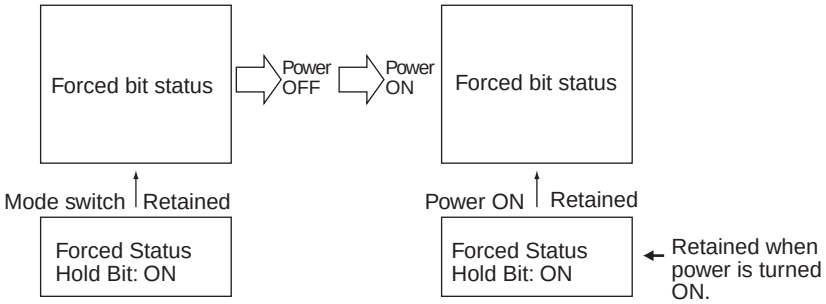
If the Forced Status Hold Bit Startup Hold parameter is set to ON, the status of the Forced Status Hold Bit will be protected when the PLC is turned ON. If this parameter is set to ON and the Forced Status Hold BIt itself is ON, all force-set and force-reset bits will retain their forced status when the PLC is turned ON.

If the memory is not retained because the battery voltage is low, the Forced Status Hold Bit will be cleared whether this parameter is set to ON or OFF.

OFF: Forced Status Hold Bit cleared at startup



ON: Forced Status Hold Bit protected at startup



● IOM Hold Bit Startup Hold Setting

Use this parameter to set whether to retain the status of the IOM Hold Bit in the Auxiliary Area at startup.

Parameter	Settings	Default	Function	Related flags and words
IOM Hold Bit Startup Hold Setting	OFF: Cleared ON: Retained	OFF	This setting determines whether the status of the IOM Hold Bit (A500.12) is retained at startup. When you want all of the data in I/O Memory to be retained when the power is turned on, turn ON the IOM Hold Bit and set this setting to ON.	A500.12 (IOM Hold Bit)

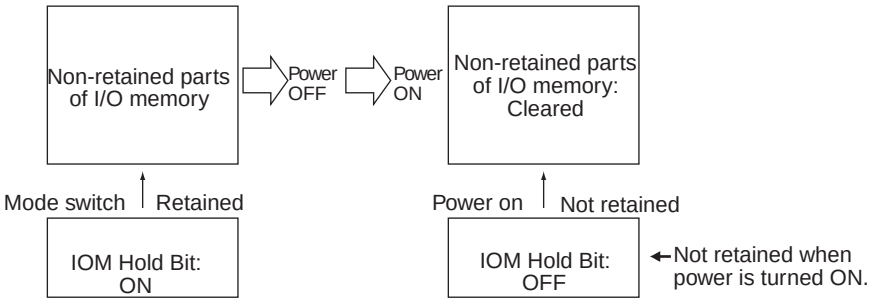
- Relation between IOM Hold Bit and this Parameter

The IOM Hold Bit (A500.12) can be turned ON to retain all of the data in I/O memory when the CPU Unit's operating mode is switched from PROGRAM mode to RUN or MONITOR mode, and vice versa. When the PLC is turned ON, the IOM Hold Bit itself will be turned OFF unless it is protected with this PLC Setup setting.

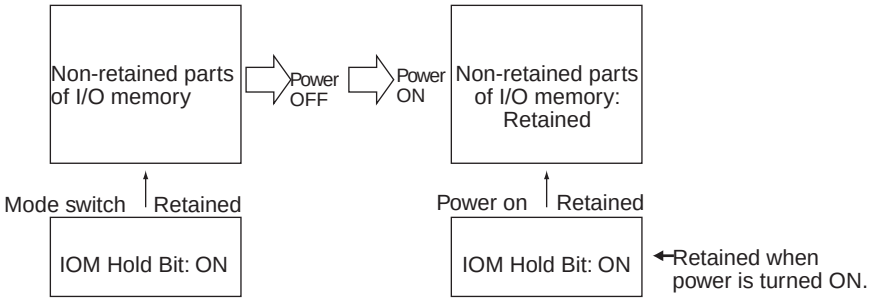
If the IOM Hold Bit Status at Startup setting is ON, the status of the IOM Hold Bit will be protected when the PLC is turned ON. If this parameter is set to ON and the IOM Hold Bit itself is ON, all data in I/O memory will be retained when the PLC is turned ON.

If the memory is not retained because the battery voltage is low, the IOM Hold Bit will be cleared whether this parameter is set to ON or OFF.

OFF: IOM Hold Bit cleared at startup



ON: IOM Hold Bit protected at startup



Mode

● PROGRAM, MONITOR, or RUN

Set the operating mode to be used at startup.

Parameter	Settings	Default	Function	Related flags and words
Mode	• Program: PROGRAM mode • Monitor: MONITOR mode • Run: RUN mode	RUN mode	Sets the CPU Unit's operating mode at startup.	---

Execution Setting

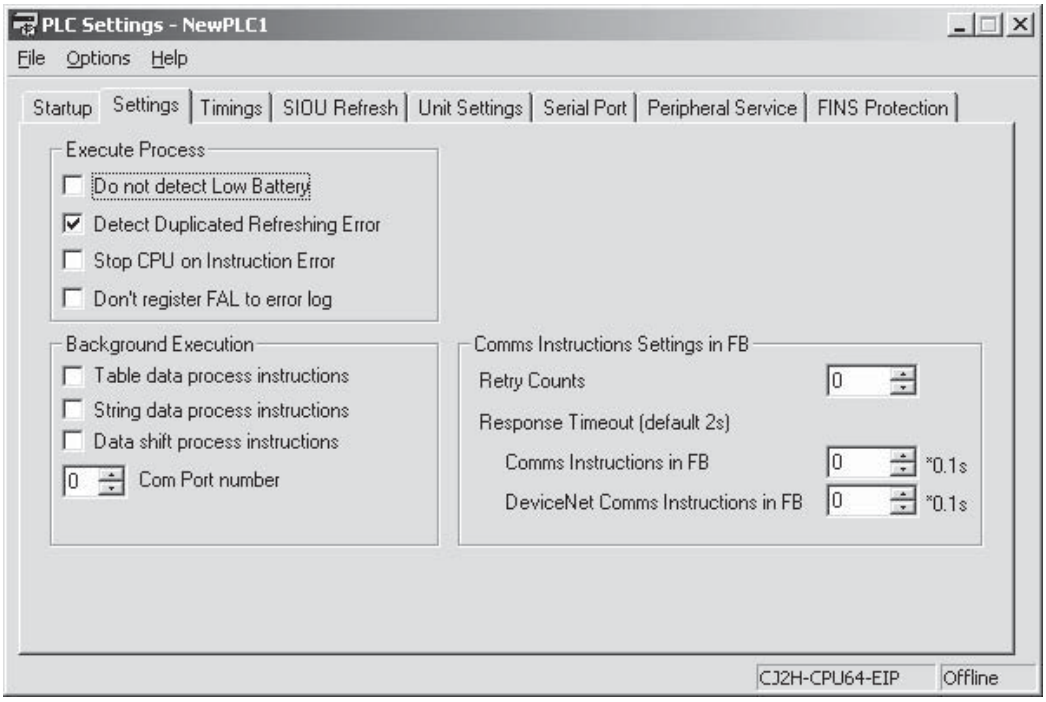
● Start Running Program when Initializing Unit/Inner Board Recognition

Set whether to wait for startup processing of specific Units at startup.

Parameter	Settings	Default	Function	Related flags and words
Start running program when initializing Unit/Inner board recognition	OFF: Wait for Units. ON: Do not wait.	OFF	To start the CPU Unit in RUN or MONITOR mode even if there is one or more Units*1 that has not completed startup processing, set this setting to ON (Don't wait for Units). To wait for all Units to finish startup processing, set this setting to OFF (Wait for Units).	---

*1 This setting applies only to specific Units. If "do not wait" is set, the CPU Unit will not wait only for those specific Units, i.e., it will still wait for all other Units to start.

9-3-2 CPU Unit Settings



Execute Process

● Do not Detect Low Battery (Operating without a Battery)

Set whether to detect battery errors (default: Detect). Use the *Do not detect* setting to operate without a battery. For details, refer to information in the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).

Parameter	Settings	Default	Function	Related flags and words
Do not detect Low Battery	OFF: Detect ON: Do not detect	OFF	This setting determines whether the CPU Unit battery errors are detected. If this setting is set to OFF (detect) and a battery error is detected, the ERR/ALM indicator on the CPU Unit will flash and the Battery Error Flag (A402.04) will be turned ON, but CPU Unit operation will continue.	A402.04 (Battery Error Flag)



Additional Information

Battery-free Operation

- **Operating without a Battery**

The following data can be held when the PLC operates without a battery.

- User program and network symbols
- Parameters (e.g., PLC Setup, registered I/O tables, routing tables, CPU Bus Unit settings, PLC names)
- Data saved in the non-volatile memory in the CPU Bus Units and Special I/O Units (e.g., protocol macro data in Serial Communications Units)

The following data is not held. The values will not be stable.

- I/O memory (including the Holding, DM, EM, and Auxiliary Areas)
- The clock built into the CPU Unit

- **Setting Operation without a Battery**
  - PLC Setup

    Set the PLC Setup as described below.

    (1) The I/O memory will be unstable when there is no battery, so clear the IOM Hold Bit Check Box so that the IOM Hold Bit is cleared at startup.

    (2) Forced status will be unstable when there is no battery, so clear the Forced Status Hold Bit Check Box so that Forced Status Hold Bit is cleared at startup.

    (3) Select the Do not detect Low Battery Check Box.
  - Programming

    The operation of the Output OFF Bit will be unstable when there is no battery, so insert the following instructions so that the Output OFF Bit does not turn ON.

    ```
    LD P_Off
    OUT A500.15
    ```

- **Precautions for Operation without a Battery**

Be careful of the following points when you operate without a battery.

- Initializing Data

  Values in the DM Area, EM Area, and other I/O memory areas will be unstable. Be sure to set the initial values from the program.

  Example:

  ```
  LD A200.11
  MOV #0918 D0 // Initialization of D0 values at start of operation
  ```

- **Unstable Clock**

The clock built into the CPU Unit will not operate and the values will be unstable. Therefore, the data on dates and times recorded in the error log will not be displayed correctly. Also, if files are saved on a Memory Card, the date and time that the file was created will not be stable.

Precautions for Correct Use

The contents of the DM, EM, HR, and AR Areas in the CPU Unit are not backed up to internal flash memory. The contents of the DM, EM, HR, and AR Areas are retained by the battery when the power is turned OFF or interrupted. This data may be lost if there is a battery error. Provide measures in the program using the Battery Error Flag (A402.04) to re-initialize data.

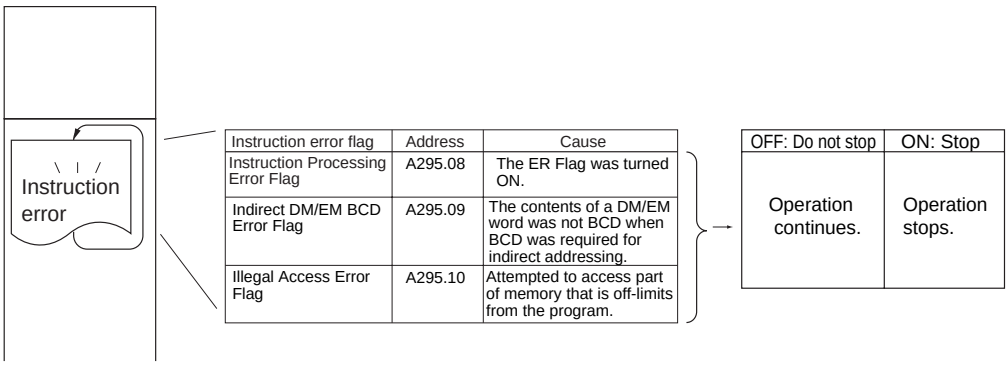
● Detect Duplicated Refreshing Error (Setting Whether to Detect Duplicate Refresh Errors)

This setting determines whether duplicate refresh errors are detected (default: detect). Duplicate refresh errors will be detected in the following cases if this parameter is set to the default (ON: detect).

Parameter	Settings	Default	Function	Related flags and words
Detect Duplicated Refreshing Error	OFF: Do not detect ON: Detect	ON	This setting determines whether duplicate refresh errors are detected. If this setting is set to ON (detect) and an error is detected, the ERR/ALM indicator on the CPU Unit will flash and the Duplicate Refresh Error Flag (A402.13) will be turned ON. CPU Unit operation will continue.	A402.13 (Duplicate Refresh Error Flag)

● Stop CPU on Instruction Error

This setting determines whether operation will be stopped if an instruction execution error occurs (default: Do not stop). Stopping the CPU Unit for instruction errors is used when debugging the program. A program error will be generated as an instruction error if any of the following flags is turned ON if Stop CPU on Instruction Error is set to stop the CPU Unit.



Parameter	Settings	Default	Function	Related flags and words
Stop CPU on Instruction Error	OFF: Continue ON: Stop	OFF	This setting determines whether instruction errors (instruction processing errors (ER) and illegal access errors (AER)) are treated as non-fatal or fatal errors.	A295.08 (Instruction Processing Error Flag), A295.09 (Indirect DM/EM BCD Error Flag, A295.10 (Illegal Access Error Flag) (If this setting is set to OFF, these flags won't be turned ON even if an instruction error occurs.)

● Don’t Register FAL to Error Log

This parameter determines whether to register the error to the error log when a user-programmed FAL error occurs.

Parameter	Settings	Default	Function	Related flags and words
Don't register FAL to error log	OFF: Record user-defined FAL errors in error log. ON: Don't record user-defined FAL errors in error log.	OFF	This setting determines if user-defined FAL errors created with FAL(006) and time monitoring for FPD(269) will be recorded in the error log (A100 to A199). Set it to ON to prevent these errors from being recorded.	---

Background Execution Settings

The following instruction will have a large affect on the cycle time, depending on the amount of data handled by the instruction. The background execution settings can be used to divide processing of the instructions over more than one cycle, to reduce the affect on the cycle time. Refer to *10-2-5 Background Execution* for details.

● Table Data Process Instructions

This parameter determines whether to process Table Data Instructions in the background.

Parameter	Settings	Default	Function	Related flags and words
Table data process instructions	OFF: Not executed in background. ON: Executed in background.	OFF	This setting determines if Table Data Instructions will be processed over multiple cycle times (i.e., processed in the background).	---

● String Data Process Instructions

This parameter determines whether to process String Data Instructions in the background.

Parameter	Settings	Default	Function	Related flags and words
String data process instructions	OFF: Not executed in background. ON: Executed in background.	OFF	This setting determines if String Data Instructions will be processed over multiple cycle times (i.e., processed in the background).	---

● Data Shift Process Instructions

This parameter determines whether to process Data Shift Instructions in the background.

Parameter	Settings	Default	Function	Related flags and words
Data shift process instructions	OFF: Not executed in background. ON: Executed in background.	OFF	This setting determines if Data Shift Instructions will be processed over multiple cycle times (i.e., processed in the background).	---

● Com Port Number

This parameter sets the communications port number used in background processing.

Parameter	Settings	Default	Function	Related flags and words
Com Port number	0 to 7: Communications ports 0 to 7 (internal logical ports)	0 (No. 0)	The communications port number (internal logical port) that will be used for background execution.	---

Precautions for Correct Use

Background processing cannot be used in interrupt tasks if high-speed interrupt function is enabled in the PLC Setup. An instruction processing error will occur.

Comms Instructions Settings in FB (for FB Library)

The following parameters are used only for OMRON function blocks. They are not used for any other application.

The number of resends and response monitoring time must be set by the user in the FB communications instructions settings in the PLC Setup, particularly when using function blocks from the OMRON FB Library to execute FINS messages or DeviceNet explicit messages communications. The values set in this PLC Setup for OMRON FB Library will be automatically stored in the related Auxiliary Area words A580 to A582 and used by the function blocks from the OMRON FB Library.

● Retry Counts

This parameter determines the number of retries for executing communications instructions within function blocks.

Parameter	Settings	Default	Function	Related flags and words
Retry Counts	0 to 15	0	Set the number of retries for sending commands when executing DeviceNet explicit messages or FINS messages within function blocks.	A580.00 to A580.03

● Response Timeout for Comms Instructions in FB

This parameter determines the response monitoring time for executing communications instructions in function blocks.

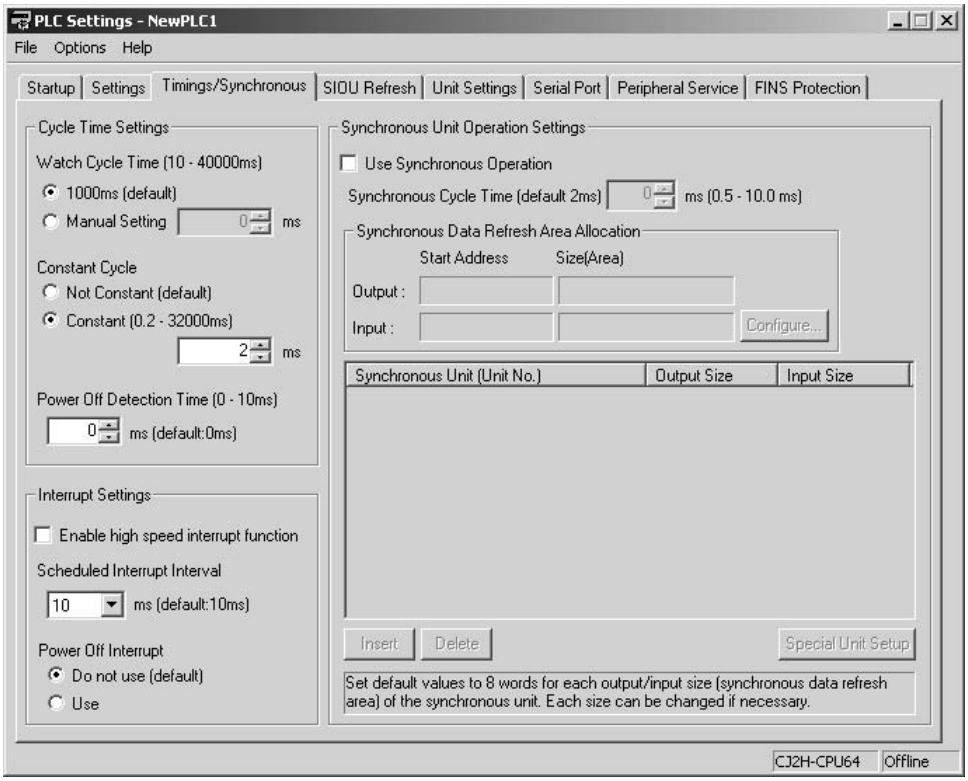
Parameter	Settings	Default	Function	Related flags and words
Comms Instructions Settings in FB	0001 to 65535 (Unit: 0.1 s, 0.1 to 6553.5) 0000: 2 s	0000: 2 s	A response timeout occurs when no response is returned within the time set here for FINS commands executed within a function block.	A581

● Response Timeout for DeviceNet Comms Instructions in FB

This parameter determines the response monitoring time for executing DeviceNet communications instructions in function blocks.

Parameter	Settings	Default	Function	Related flags and words
DeviceNet Comms Instructions in FB	0001 to FFFF (Unit: 0.1 s, 0.1 to 6553.5) 0000: 2 s	0000: 2 s	A response timeout occurs when no response is returned within the time set here for explicit messages commands executed within a function block.	A582

9-3-3 Timings/Synchronous Settings



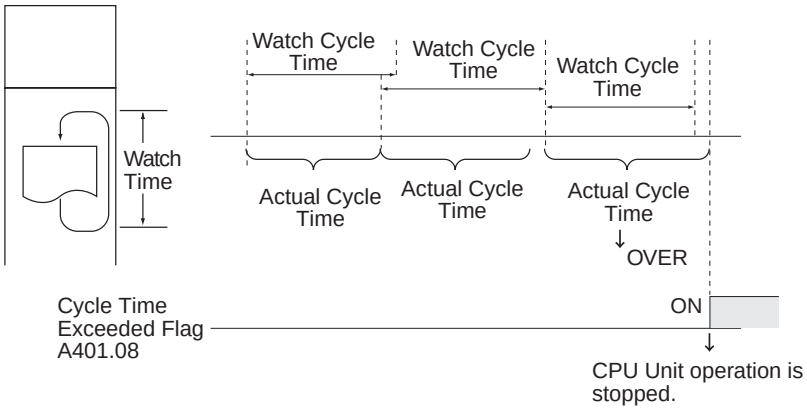
● Watch Cycle Time

This parameter is used to set the Watch Cycle Time to settings other than the default (1000 ms).

Parameter	Settings	Default	Function	Related flags and words
Watch Cycle Time	ON: Default (1,000 ms: 1 s) OFF: Manual Setting	ON (1,000 ms: 1 s)	Set to OFF to set any watch cycle time other than the default value of 1 s.	A401.08 (Cycle Time Exceeded Flag)
	10 to 40,000 ms (10-ms units)	0	This setting is valid only when the above parameter is set to OFF (manual setting). If the cycle time exceeds the value set for the watch cycle time, A401.08 (Cycle Time Too Long Flag) will turn ON.	A264 and A265 (Present Cycle Time)

- When to Change the Watch Cycle Time

  If the cycle time exceeds the watch (maximum) cycle time setting, the Cycle Time Exceeded Flag (A401.08) will turn ON and PLC operation will be stopped. This parameter must be changed if the normal cycle time exceeds the default watch cycle time setting of 1 s.



Note The default value for the watch cycle time is 1 s (1,000 ms).

● Cycle Time

This parameter sets the minimum cycle time when the minimum cycle time function is used (default: variable cycle time).

Parameter	Settings	Default	Function	Related flags and words
Constant Cycle Time	OFF: Minimum cycle time ON: Variable	ON (variable)	Set this setting to OFF to use a minimum cycle time. If a minimum cycle time is to be used, the cycle time must be set.	---
	0.2 to 32,000 ms (0.1-ms units)	---	Set to 0.2 to 32000.0 to specify a minimum cycle time. If the cycle time is less than this setting, it will be extended until this time passes. Leave this setting at 0 for a variable cycle time.	---

Additional Information

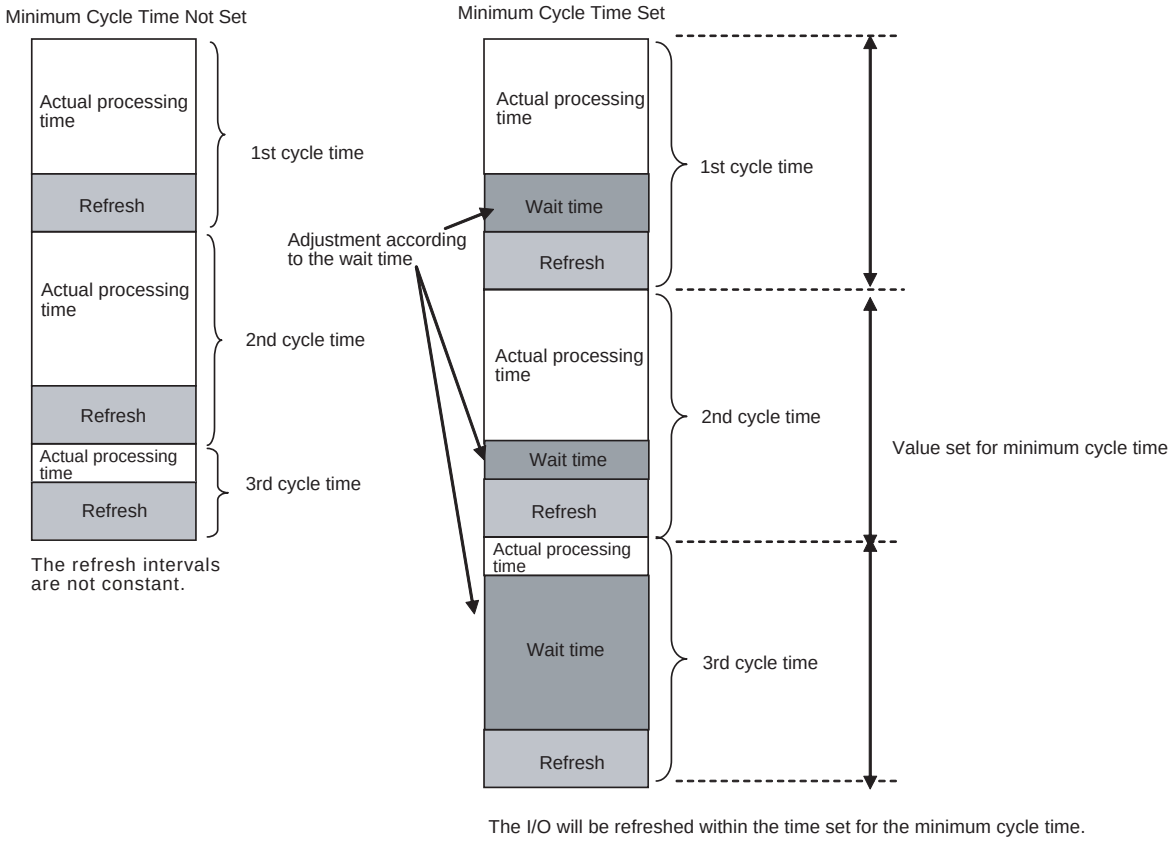
The minimum cycle time can be changed from the CX-Programmer display to monitor the cycle time when the CPU Unit is in MONITOR mode (unit version 1.1 or later only).

Precautions for Correct Use

If a longer cycle time is set, the service interval for Support Software will also increase, which may result in poor responsive of online operations and difficulty in connecting online.

- Conditions for Enabling a Minimum Cycle Time

  Set the minimum cycle time to a non-zero value to eliminate inconsistencies in I/O responses. This parameter is effective only when the actual cycle time is shorter than the minimum cycle time setting. If the actual cycle time is longer than the minimum cycle time setting, the actual cycle time will remain unchanged.



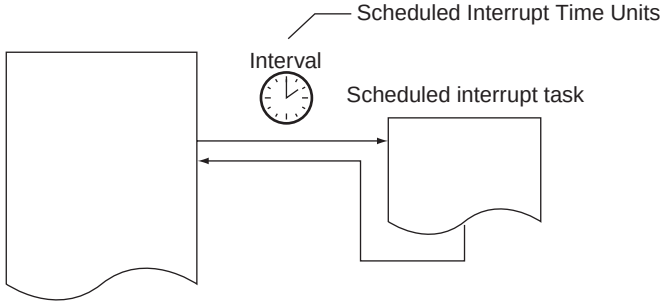
● Scheduled Interrupt Interval

This parameter sets the time unit used in scheduled interrupt intervals to 1 ms, 0.1, or the default of 10 ms.

Parameter	Settings	Default	Function	Related flags and words
Scheduled Interrupt Interval	• 10 ms • 1.0 ms • 0.1 ms	10 ms	This setting determines the time units used in scheduled interrupt interval settings.	---

- Scheduled Interrupt Time Unit

  This parameter sets the time unit for the scheduled interrupt interval settings. Set the scheduled interrupt interval from the program with MSKS(690).



● Power OFF Detection Delay Time

This parameter is used to set the power OFF detection delay time to a value other than 0 ms.

Parameter	Settings	Default	Function	Related flags and words
Power Off detection time	0 to 10 ms (1-ms units)	0 ms	This setting determines how much of a delay there will be from the detection of a power interruption (approximately 10 to 25 ms for AC power and 2 to 5 ms for DC power after the power supply voltage drops below 85% of the rated value) to the confirmation of a power interruption. The default setting is 0 ms. When the power OFF interrupt task is enabled, it will be executed when the power interruption is confirmed. If the power OFF interrupt task is disabled, the CPU will be reset and operation will be stopped.	---

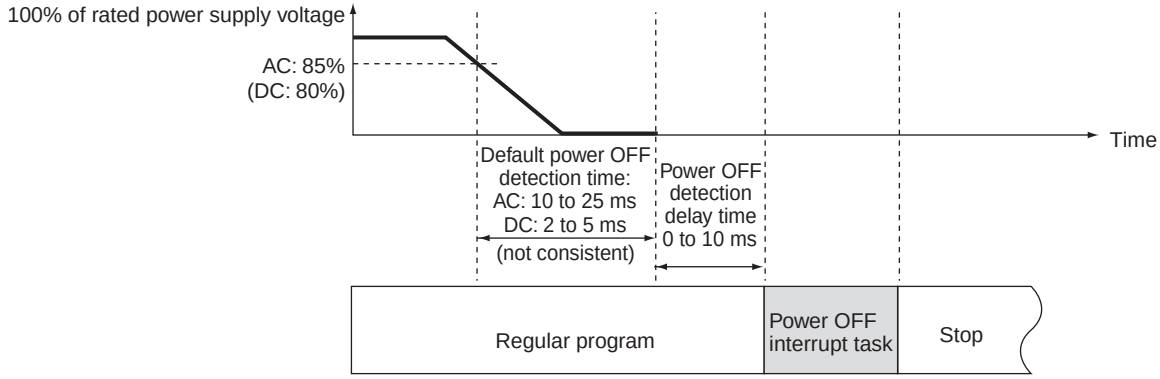
Note This parameter is not supported when the CJ1W-PD022 Power Supply Unit is mounted.

Power OFF Detection Delay Time

This parameter determines how much of a delay there will be from the detection of a power interruption until a power interruption is established and the regular program is stopped. The setting can be between 0 and 10 ms.

Extend the time until detection of a power interruption when momentary interruptions in a bad power supply are causing PLC operation to stop. It takes a maximum of 10 ms for the internal 5-VDC power supply to drop to 0 VDC after the initial power interrupt detection time. This 10 ms is the power supply holding time. It is equal to the power OFF detection time plus the processing time required to confirm the power interruption. The power OFF delay time is 10 to 25 ms (not consistent) for AC power supplies, 2 to 5 ms for CJ1W-PD025 DC Power Supply Units, and 2 to 10 ms for CJ1W-PD022 DC Power Supply Units.

Note This parameter is not supported when the CJ1W-PD022 Power Supply Unit is mounted. (Refer to *A-5 Operation for Power Interruptions*.)



Note The execution time of the program in the power OFF interrupt task must be less than 10 ms minus the power OFF detection delay time. For information on power OFF interrupt tasks, refer to *5-2-3 Interrupt Tasks.* For information on operation at power OFF, refer to *A-5 Operation for Power Interruptions*.

● Power OFF Interrupt Disabled

This parameter determines whether the power OFF interrupt task is enabled.

Parameter	Settings	Default	Function	Related flags and words
Power Off Interrupt disabled	ON: Disabled OFF: Enabled	ON	When this setting is set to OFF (enabled), the power OFF interrupt task will be executed when power is interrupted.	---

Note This parameter is not supported when the CJ1W-PD022 Power Supply Unit is mounted.

● Power OFF Interrupt Task

This parameter determines whether a power OFF interrupt task will be executed when a power interruption is detected. (When this parameter is set to ON, the regular program will just stop when a power interruption is detected.)

The power OFF interrupt task will be stopped when the power hold time (processing time after power interrupt + power OFF detection delay time) has elapsed. The maximum power hold time is 10 ms.

When a power OFF detection delay time has to be set, be sure that the power OFF interrupt task can be executed in the available time (10 ms minus the power OFF detection delay time).

Note This parameter is not supported when the CJ1W-PD022 Power Supply Unit is mounted. (Refer to *A-5 Operation for Power Interruptions*.)

● Enable High-speed Interrupt Function (CJ2H CPU Units Only)

This parameter is used to enable or disable high-speed interrupt function.

Parameter	Settings	Default	Function	Related flags and words
Enable high-speed interrupt function	OFF: Disable high-speed interrupt function. ON: Enable high-speed interrupt function.	OFF (Disable high-speed interrupts.)	High-speed interrupt function is enabled when the *Enable high-speed interrupt function* Option is selected.	---

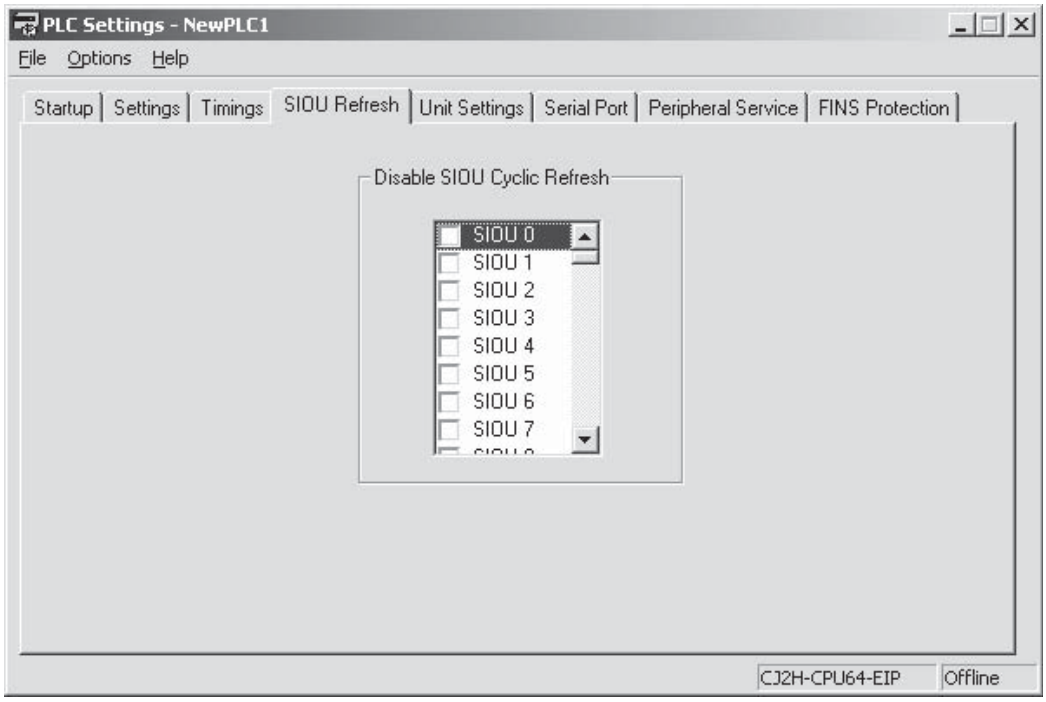
High-speed interrupt function improve execution of interrupt tasks under certain restrictions. For details, refer to *10-2-6 High-speed Interrupt Function*.

If the *Use Synchronous Operation* Check Box is selected in the PLC Setup, the *Enable high-speed interrupt function* Check Box will be selected automatically.

● Synchronous Unit Operation Settings (CJ2H CPU Units Only)

These settings are required to use the synchronous unit operation function. Refer to *10-8-7 PLC Setup* for details.

9-3-4 Special I/O Unit Cyclic Refreshing



Disable SIOU Cyclic Refresh

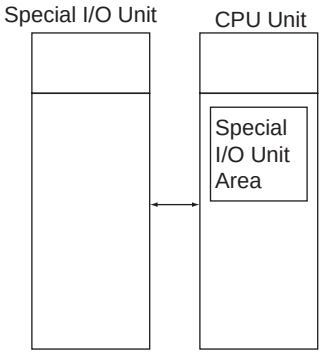
These parameters specify whether to disable cyclic refreshing for Special I/O Units (SIOU).

Parameter	Settings	Default	Function	Related flags and words
Cyclic Refreshing for Units 0 to 95 There is a parameter for each group of 15 Units	OFF: Enabled ON: Disabled	OFF (disabled)	These settings determine whether data will be exchanged between the specified Unit and the Special I/O Unit's allocated words (10 words/Unit) during cyclic refreshing for Special I/O Units.	---

● Prohibiting Cyclic Refreshing of Special I/O Units:

Always disable cyclic refreshing of the Special I/O Units if an IORF(097), FIORF(225), IORD(222), or IOWR(223) instruction is to be used to refresh the Special I/O Units in an interrupt task. If any of the following is executed in an interrupt task when cyclic refreshing is enabled for the Special I/O Units a duplicated refreshing error (non-fatal) will occur, and the Duplicate Error Refresh Flag (A402.13) will turn ON.

- Refreshing I/O for the same Special I/O Unit with an IORF(097)/FIORF(225) instruction
- Reading or writing data for memory areas in the same Special I/O Unit with an IORD(222)/IOWR(223) instruction

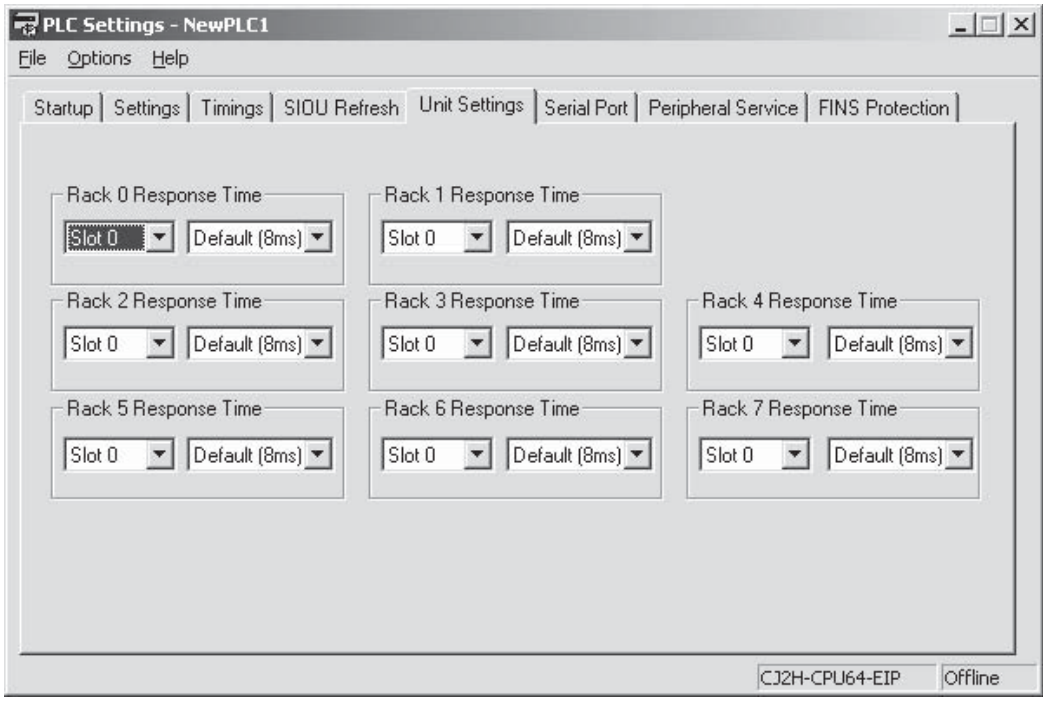


The settings in the CPU Unit determine whether data will exchange data with the specified Special I/O Units for the 10 words allocated to each Special I/O Unit in the CPU Unit (between CIO 2000 to CIO 2959) during cyclic I/O refreshing.

Precautions for Correct Use

Whenever disabling a Special I/O Unit's cyclic refreshing, be sure that the I/O for that Unit is refreshed with IORF(097) or FIORF(225) in the program at least every 11 seconds during operation. A CPU Unit service monitoring error will occur in the Special I/O Unit if it is not refreshed every 11 seconds. (The ERH indicator and the RUN indicator on the Special I/O Unit will light.)

9-3-5 Basic I/O Unit Rack Response Times



Rack 0 to 7 Rack Response Times

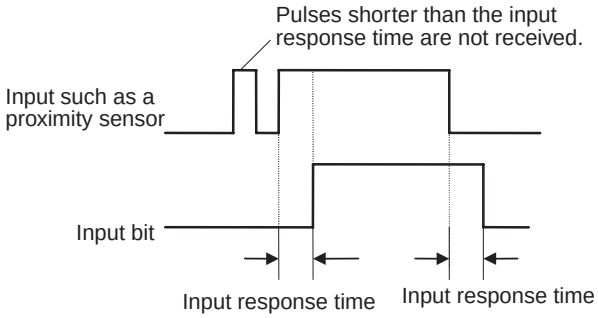
These parameters are used to set the input response times for Basic I/O Units to settings other than the default of 8 ms.

The power supply to the PLC must be turned OFF and then ON after transferring the PLC Setup to the CPU Unit.

Parameter	Settings	Default	Function	Related flags and words
Rack 0, Slots 0 to 9 Rack 1, Slots 0 to 9 Rack 2, Slots 0 to 9 Rack 3, Slots 0 to 9 Rack 4, Slots 0 to 9 Rack 5, Slots 0 to 9 Rack 6, Slots 0 to 9 Rack 7, Slots 0 to 9	Default: 8 ms No filter 0.5 ms 1 ms 2 ms 4 ms 8 ms 16 ms 32 ms	8 ms	Sets the input response time (ON response time = OFF response time) for CJ-series Basic I/O Units. If no filter is set (0 ms), there will be a delay in the ON or OFF response time based on delays in the Unit's internal elements. For details on delays in ON/OFF response time for each Unit, refer to the *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472).	A220 to A259: Actual input response times for Basic I/O Units

Changing the Basic I/O Unit Rack Response Time

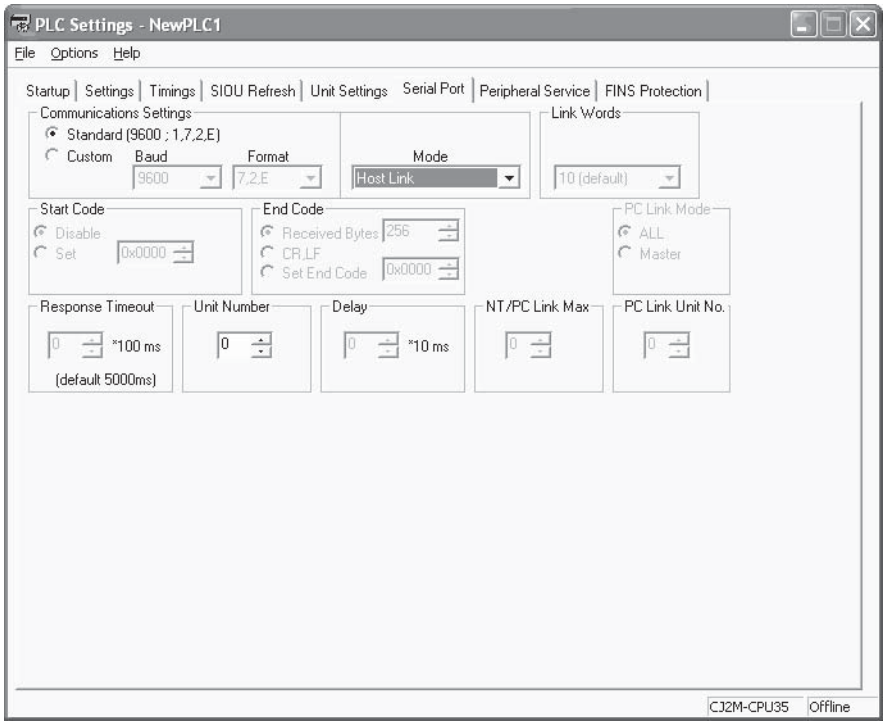
The Rack response time can be set for Basic I/O Units by Rack and slot. Increasing a setting reduces the effects of chattering and noise. Decreasing a setting allows reception of shorter input pulses. Do not set the ON response time or OFF response time to less than the cycle time.



The Rack response time settings are transferred to the Basic I/O Units when the PLC is turned ON.

When the Unit's settings are changed, they are stored in A220 to A259 (Actual Input Response Times for Basic I/O Units). When the settings in the PLC Setup have been changed with the PLC in PROGRAM mode, the PLC Setup settings will differ from the actual settings in the Units. In this case, the values in A220 to A259 can be checked to see the input response times actually set in the Units.

9-3-6 Serial Port Settings



The following parameters are valid when pin 5 on the DIP switch on the CPU Unit is OFF (default).

Communications Settings

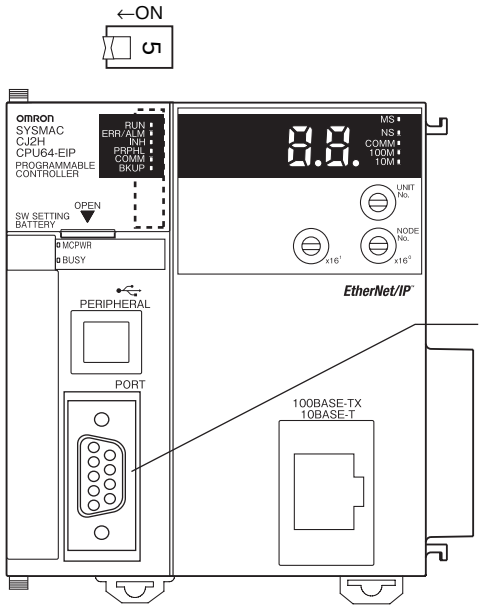
Parameter	Settings	Default	Function	Related flags and words
Communications Settings	• Standard (9600: 1, 7, 2, E): Standard settings • Custom: Any setting	Standard	The standard settings (9600: 1, 7, 2, E) are for Host Link Mode, 1 start bit, 7 data bits, even parity, 2 stop bits, and a baud rate of 9,600 bps. Set custom settings to use any other communications settings.	A619.02 (Serial Port Settings Changing Flag)

- Serial Port Settings

  Set the serial port settings in the PLC Setup when you need to change these settings from the defaults. Specify the frame format when no-protocol mode is selected.

  The port settings can also be changed with STUP(237). The Serial Port Settings Changing Flag (A619.02) is turned ON when STUP(237) is executed and it is turned OFF when the port settings have been changed.

  When pin 5 of the DIP switch on the front of the CPU Unit is ON, the CPU Unit automatically detects the communications parameters of a Programming Device (including a Programming Console) connected to the RS-232C port. Those automatically detected parameters are not stored in the PLC Setup.

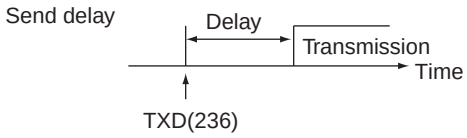


Serial port communications settings when DIP switch pin 5 is OFF:

Default settings:
Host link mode, 1 start bit, 7 data bits, even parity, 2 stop bits, and a baud rate of 9,600 bps

Custom settings:
Set the communications mode (host link, NT Link, no-protocol*, or peripheral bus) and other settings, such as the baud rate.

A send delay can be set in no-protocol mode. The operation of this delay is shown in the following diagram.



The following table shows the message formats that can be set for transmissions and receptions in no-protocol mode.

		End code setting		
		None	**Yes**	**CR+LF**
Start code setting	None	DATA	DATA ED	DATA CR+LF
	Yes	ST DATA	ST DATA ED	ST DATA CR+LF
No. of bytes received		Data: 1 to 256 bytes		

Selecting *Standard (9600: 1,7,2,E)*

If *Standard (9600: 1,7,2,E)* is selected, the following communications settings will be used unconditionally.

- Baud rate: 9,600 bps
- Start bits: 1 bit
- Data length: 7 bits
- Stop bits: 2 bits
- Parity: Even
- Mode (serial communications mode): Host link

Note The mode and other settings will be disabled.

Use the following procedure to set the communications to custom settings.

1 Select the *Custom* Option for the communications settings.

2 Select the serial communications mode in the *Mode* Field.

3 Make other settings.

Selecting *Custom*

If custom settings is selected, the following baud rates, parameters, and modes can be selected.

Baud Rate (When Custom Settings Is Selected)

Setting	Meaning	Setting	Meaning	Related flags and words
115200	11,5200 bps	4800	4,800 bps	A619.02 (Serial Port Settings Changing Flag)
57600	57,600 bps	2400	2,400 bps	
38400	38,400 bps	1200	1,200 bps	
19200	19,200 bps	600	600 bps	
9600 (default)	9,600 bps	300	300 bps	

● Format (When Custom Settings Is Selected)

Selected data	Data length	Stop bits	Parity	Related flags and words
7,1,E	7	1 bit	Even	A619.02 (Serial Port Settings Changing Flag)
7,1,O			Odd	
7,1,N			None	
7,2,E (default)		2 bits	Even	
7,2,O			Odd	
7,2,N			None	
8,1,E	8	1 bit	Even	
8,1,O			Odd	
8,1,N			None	
8,2,E		2 bits	Even	
8,2,O			Odd	
8,2,N			None	

Mode (When Custom Settings Is Selected)

Select the serial communications mode for the built-in RS-232C port.

Parameter	Settings	Default	Function	Related flags and words
Mode	• Host link • NT link (1:N) • RS-232C (no-protocol) • Peripheral Bus (Toolbus) • Serial gateway • Serial PLC Link (Polling Unit) • Serial PLC Link (Polled Unit)	Host link*1	This setting determines whether the serial port will operate in host link mode or another serial communications mode.*2	A619.02 (Serial Port Settings Changing Flag)

*1 The host link (SYSMAC WAY) is the communications mode for connection with a general host computer.

*2 Communications will not be possible with PTs set for 1:1 NT Links.

● Related Settings for Each Selection Mode

Mode	Related settings
Host link (default)	Baud rate, parameter, unit number
NT link (1:N)	Baud rate, NT/PC link max.
RS-232C (No-protocol)	Baud rate, parameter, start code, end code, delay
Peripheral Bus (Tool Bus)	Baud rate
Serial gateway	Baud rate, parameter, response monitoring time
Serial PLC Link (Polling Unit)	Communications settings, mode, number of link words, link method, maximum unit number in NT Link or Serial PLC Link (No. NT/PC Link Max.)
Serial PLC Link (Polled Unit)	Communications settings, mode, Serial PLC Link Polled Unit unit number

● Host Link Settings

Use the Host Link Mode to perform communications using Host Link with a computer or other host device and also when a baud rate other than 9,600 bps will be used.

- Unit Number (for CPU Unit in Host Link Mode)

Set the unit number for a PLC slave.

Parameter	Settings	Default	Function	Related flags and words	Update timing in CPU Unit
Unit Number	0 to 31	0	This setting determines the CPU Unit's unit number when it is connected in a 1-to-N (N=2 to 32) Host Link.	A619.02 (Serial Port Settings Changing Flag)	Each cycle

● NT Link Settings

Use the NT Link Mode to perform communications with an OMRON Programmable Terminal (NS Series or NT Series).

Baud Rate (bps)

Parameter	Settings	Default	Function	Related flags and words
Baud Rate	115,200, 38,400 bps	115,200 bps	It is recommended to set to 115,200 bps when setting this value from the CX-Programmer.	A619.02 (Serial Port Settings Changing Flag)

NT/PC Link Max.

Parameter	Settings	Default	Function	Related flags and words
NT/PC Link Max.	0 to 7	1	This setting determines the highest unit number of PT that can be connected to the PLC.	A619.02 (Serial Port Settings Changing Flag)

● RS-232C (No-protocol) Settings

Use the No-protocol Mode to perform no-protocol communications with a bar coder or other device.

Start Code

Parameter	Settings	Default	Function	Related flags and words
Start Code	Disable or Set	Disables	Set whether a start code will be used for the frame format of data to be sent and received with no-protocol communications.	---
	0x0000 to 00FF (0000 to 00FF hex: “0x” means the same as hex.)	0x0000 (00 hex)	Set the start code as a hexadecimal value.	

End Code

Parameter	Settings	Default	Function	Related flags and words
End Code	• Received Bytes (no end code) • CR LF • Set End Code	Received Bytes	Set the type of end code that will be used for the frame format of data to be sent and received with no-protocol communications	---
Received Bytes	256 or 1 to 255	256	If the number of received bytes is selected for the end code, the data length will be set in byte units for no-protocol communications sent and received. The length of the end code and start code are not included in the data length. The maximum data length that can be sent or received with one TXD(236)/RXD(235) instruction is 256 bytes by default.	
Set End Code	0x0000 to 00FF (0000 to 00FF hex: “0x” means the same as hex.)	0x00 (00 hex)	If “end code” is specified for the end code, set it here as a hexadecimal value.	

Delay

Parameter	Settings	Default	Function	Related flags and words
Delay	0 to 99990 ms (10-ms units)	0 ms	When the TXD(236) instruction is executed, data will be sent from the serial port after the send delay set here has expired.	---

● Peripheral Bus (Toolbus) Settings

Select the Peripheral Bus Mode to perform communications with the CX-Programmer or other Programming Device.

Baud Rate

Parameter	Settings	Default	Function	Related flags and words
Baud rate	9600, 19200, 38400, 57600, 115200 bps	115200 bps	The peripheral bus (toolbus) is specified.	A619.02 (Serial Port Settings Changing Flag)

It is possible to automatically detect the baud rate at the CX-Programmer and connect the Peripheral Bus by setting DIP switch pin 5 on the CPU Unit to ON.

● Serial Gateway Settings

The Serial Gateway protocol is selected to perform communications with OMRON components using CompoWay/F.

Response Monitoring Time

Parameter	Settings	Default	Function*1	Related flags and words
Response Monitoring Time	5000 ms, 100 to 25500 ms (100-ms units)	0 (50000 ms)	Monitors the time from when the FINS command that has been converted into the specified protocol using Serial Gateway is sent until the response is received. Default: 5 s; PLC Setup: 0.1 to 25.5 s	A619.02 (Serial Port Settings Changing Flag)

*1 If a timeout occurs, the FINS end code 0205 hex (response timeout) will be returned to the FINS source.

● Serial PLC Link Polling Unit Settings (CJ2M CPU Units Only)

Select a Serial PLC Link Polling Unit to enable exchanging data between CJ2M CPU Units or between CJ2M CPU Units and CJ1M/CP1H/CP1L/CP1E CPU Units without special programming.

Communications Settings

Parameter	Settings	Default	Function	Related flags and words
Baud rate	38400, 115200 bps	115200 bps	Select the baud rate when specifying the Serial PLC Link Polling Unit.	A619.02 (Serial Port Settings Changing Flag)

Number of Link Words

Parameter	Settings	Default	Function	Related flags and words
Link Words	1 to 10 words	10 words	This parameter is set only in the Polling Unit. Set the number of words used per node in the Serial PLC Link Area.	A619.02 (Serial Port Settings Changing Flag)

PLC Link Method

Parameter	Settings	Default	Function	Related flags and words
Link Method	Complete link method or Polling Unit link method	Complete Link	This setting specifies the link method for the Serial PLC Link. This parameter is set only in the Polling Unit.	A619.02 (Serial Port Settings Changing Flag)

Highest Unit Number for NT/Serial PLC Link

Parameter	Settings	Default	Function	Related flags and words
No. NT/PC Link Max.	0 to 7	0	This setting determines the highest unit number of the Polled Units connected to the Polling Unit when 1:N connections are used for Serial PLC Links. This parameter is set only in the Polling Unit.	A619.02 (Serial Port Settings Changing Flag)

● Serial PLC Link Polled Unit Settings (CJ2M CPU Units Only)

Select a Serial PLC Link Polled Unit to enable exchanging data between CJ2M CPU Units or between CJ2M CPU Units and CJ1M/CP1H/CP1L/CP1E CPU Units without special programming.

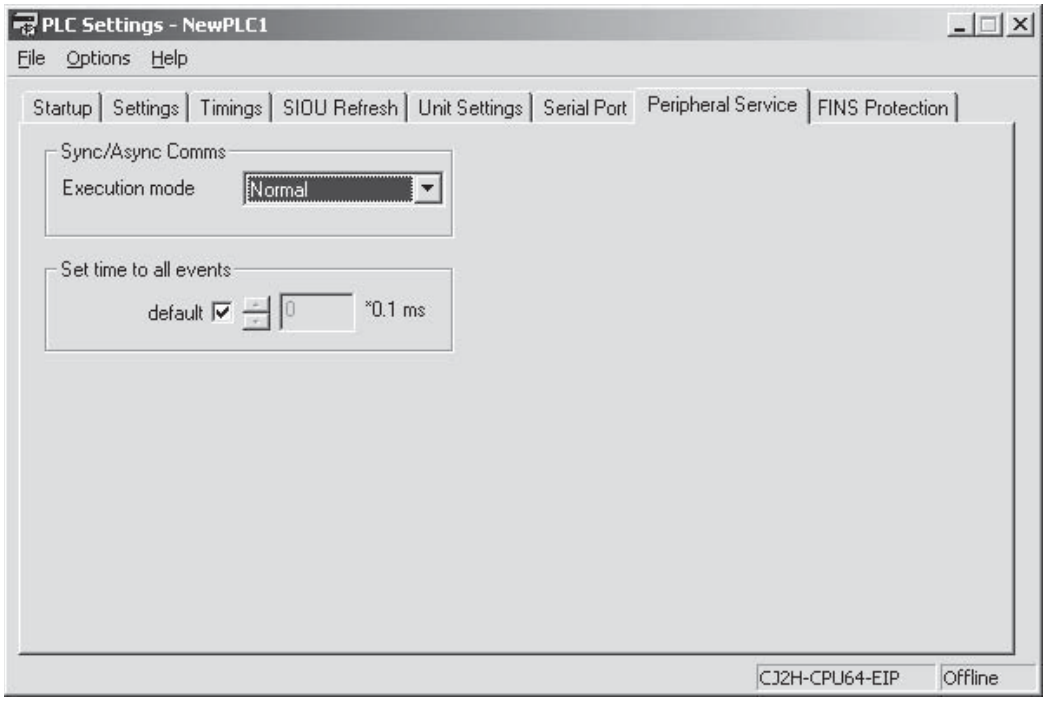
Communications Settings

Parameter	Settings	Default	Function	Related flags and words
Baud rate	38400, 115200 bps	115200 bps	Select the baud rate when specifying a Serial PLC Link Polled Unit.	A619.02 (Serial Port Settings Changing Flag)

Serial PLC Link Polled Unit Unit Number

Parameter	Settings	Default	Function	Related flags and words
Serial PLC Link Unit No.	0 to 7	0	Set the unit number of a Polled Unit connected to the Polling Unit when 1:N connections are used for Serial PLC Links.	A619.02 (Serial Port Settings Changing Flag)

9-3-7 Peripheral Service



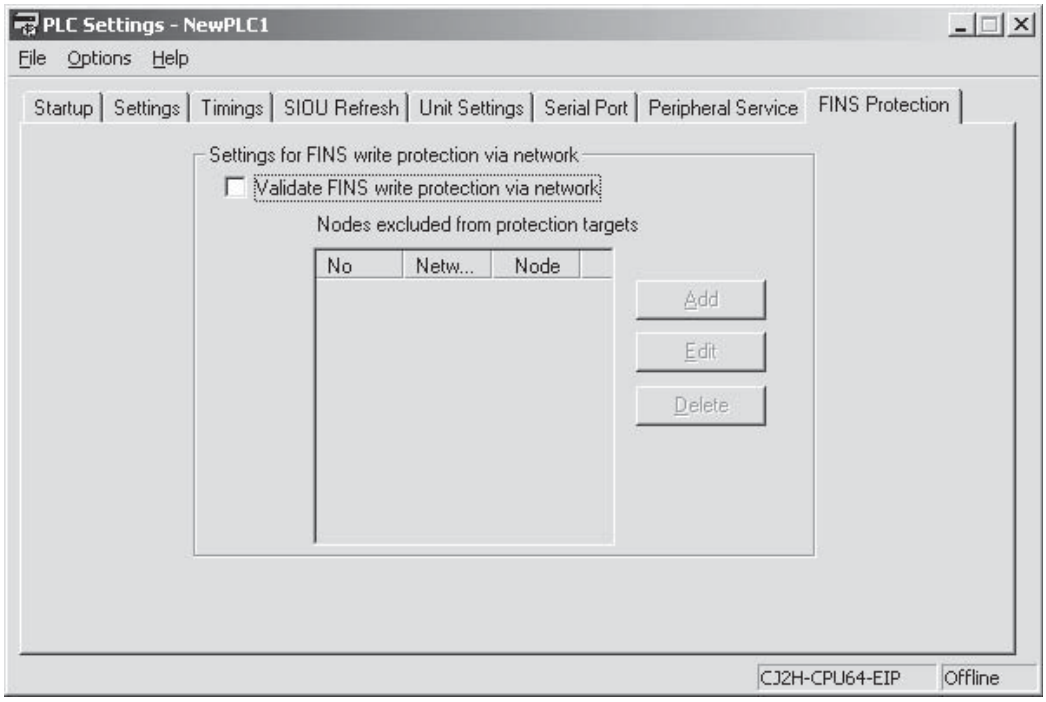
● Sync/Async Comms (CPU Processing Mode)

Parameter	Settings	Default	Function	Related flags and words
Execution mode	Normal mode	Normal Mode	The CJ2 CPU Units support only Normal Mode.	---

● Set Time to All Events

Parameter	Settings	Default	Function	Related flags and words
Set time to all events	Default: 10% of cycle time 0.1 to 3,276.7 ms	10% of cycle time (or 0.1 ms if 10% of the cycle time is less than 0.1 ms)	Sets the maximum time that will be used for all peripheral servicing. The maximum time can be set to 0.1 ms or higher.	---

9-3-8 FINS Protection



Settings for FINS Write Protection via Network

● Validate FINS Write Protection via Network

This parameter determines whether write protection is used for FINS commands over a network.

This parameter is used to prevent accidentally writing memory via a network connected with a connection other than a serial connection. If this parameter is enabled, writing will be possible only from nodes set for the *Nodes excluded from protection targets* parameter.

Parameter	Setting	Default	Function	Related flags and words
Validate FINS write protection via network	ON: Disable FINS write protection OFF: Enable FINS write protection	OFF	Enables or disables write protection for the CPU Unit from FINS command sent over a network (i.e., all connections except for serial connections).	---

● Nodes Excluded from Protection Targets

Set the nodes for which writing will be enabled even when protection is enabled.

Set the nodes and networks from which FINS write operations will be enabled even when protection is enabled. The total number of nodes set to be excluded from write protection will be automatically set.

A maximum of 32 nodes can be set. If these settings are not made (i.e., if the total number of nodes is 0), write operations will be disabled for all nodes but the local node

Parameter	Settings	Default	Function	Related flags and words
Nodes excluded from protection targets*1	0 to 127	---	FINS command source network address	---
	1 to 255*2	---	FINS command source node address	---

*1 This setting is valid only when FINS write protection has been enabled.

*2 255 (FF hex) can be set to include all nodes in the specified network.

10

CPU Unit Functions

This section describes the functions that are built into the CPU Unit.

10-1-1 Clock Functions 10-3
10-1-2 Times Stored in Memory 10-4
10-1-3 Free-running Timers 10-6

10-2 Cycle Time/High-speed Processing 10-7
10-2-1 Minimum Cycle Time 10-7
10-2-2 Maximum Cycle Time 10-8
10-2-3 Monitoring the Cycle Time 10-9
10-2-4 High-speed Inputs 10-9
10-2-5 Background Execution 10-10
10-2-6 High-speed Interrupt Function 10-19

10-3 Startup Settings and Maintenance 10-22
10-3-1 Holding Settings for Operating Mode Changes and at Startup 10-22
10-3-2 Power OFF Detection Delay Setting 10-24
10-3-3 Disabling Power OFF Interrupts 10-25
10-3-4 RUN Output 10-26
10-3-5 Automatic Transfer at Startup 10-27

10-4 Unit Management Functions 10-35
10-4-1 Basic I/O Unit Management 10-35
10-4-2 CPU Bus Unit Flags/Bits 10-37
10-4-3 Special I/O Unit Flags/Bits 10-38

10-5 Memory Management Functions 10-39
10-5-1 Automatic Backup 10-39
10-5-2 EM File Memory Functions 10-41
10-5-3 Comment Memory 10-42
10-5-4 Replacing the Entire Program during Operation 10-43

10-6 Security Functions 10-50
10-6-1 Write-protection Using the DIP Switch 10-50
10-6-2 Read Protection Using Passwords 10-50
10-6-3 Program Operation Protection Using Production Lot Numbers 10-55
10-6-4 Write Protection from FINS Commands 10-56
10-6-5 PLC Names 10-60
10-7 Debugging 10-63
10-7-1 Forced Set/Reset 10-63
10-7-2 Test Input 10-64
10-7-3 Differential Monitoring 10-64
10-7-4 Online Editing 10-65
10-7-5 Turning OFF Outputs 10-67
10-7-6 Tracing Data 10-68
10-7-7 Storing the Stop Position at Errors 10-73
10-7-8 Failure Alarm Instructions 10-74
10-7-9 Simulating System Errors 10-75
10-7-10 Failure Point Detection 10-76
10-8 Synchronous Unit Operation 10-78
10-8-1 Overview 10-78
10-8-2 Details on Synchronous Unit Operation 10-81
10-8-3 Synchronous Unit Operation Specifications 10-84
10-8-4 Synchronous Data Refresh 10-85
10-8-5 Restrictions in Using Synchronous Unit Operation 10-89
10-8-6 Application Procedure 10-91
10-8-7 PLC Setup 10-92
10-8-8 Writing the Synchronous Interrupt Task 10-94
10-8-9 Adjusting and Troubleshooting Synchronous Unit Operation 10-95

10-1 Clock Functions

10-1-1 Clock Functions

Clock Data

A clock is built into the CJ2 CPU Units.

The clock data from the clock in the CPU Unit is stored in the following bits in the Auxiliary Area in BCD.

Name	Address	Description
Clock Data	A351.00 to A351.07	Seconds: 00 to 59 (BCD)
	A351.08 to A351.15	Minutes: 00 to 59 (BCD)
	A352.00 to A352.07	Hour: 00 to 23 (BCD)
	A352.08 to A352.15	Day of the month: 01 to 31 (BCD)
	A353.00 to A353.07	Month: 01 to 12 (BCD)
	A353.08 to A353.15	Year: 00 to 99 (BCD)
	A354.00 to A354.07	Day of the week (00 to 06 BCD): 00: Sunday, 01: Monday, 02: Tuesday, 03: Wednesday, 04: Thursday, 05: Friday, 06: Saturday

If the battery is not connected or the battery voltage is low, the clock data in words A351 to A354 will be unstable. Do not use the clock data unless a reliable battery is connected.

Setting Clock Data

The data can be set by connecting the CX-Programmer online and double-clicking the PLC Clock Icon in the project tree or by executing the DATE(735) instruction. Also, clock data can be incremented and decremented and converted between seconds and hours.

● Clock Instructions

Instruction	Mnemonic	Function
CLOCK ADJUSTMENT	DATE(735)	Changes the internal clock setting to the setting in the specified source words.
HOURS TO SECONDS	SEC(065)	Converts time data in hours/minutes/seconds format to an equivalent time in seconds only.
SECONDS TO HOURS	HMS(066)	Converts seconds data to an equivalent time in hours/minutes/seconds format.
CALENDAR ADD	CADD(730)	Adds time to the calendar data in the specified words.
CALENDAR SUBTRACT	CSUB(731)	Subtracts time from the calendar data in the specified words.

10-1-2 Times Stored in Memory

Words in the Auxiliary Area automatically store the times when the power is turned ON, the times when the power is turned OFF, the number of power interruptions, the total power ON time, the times when the user memory (programs and parameters) is written, and the times when operation is started and stopped. The times stored in memory must not be used if the battery is not connected or if the battery voltage is low.

Power ON Clock Data

The year, month, day, and time that the PLC is turned ON is stored in the following Auxiliary Area words.

Name	Address	Description
Power ON Clock Data 1	A720 to A722	The data is BCD. A720.00 to A720.07: Seconds (00 to 59) A720.08 to A720.15: Minutes (00 to 59) A721.00 to A721.07: Hour (00 to 23) A721.08 to A721.15: Day of month (01 to 31) A722.00 to A722.07: Month (01 to 12) A722.08 to A722.15: Year (00 to 99)
Power ON Clock Data 2	A723 to A725	These words contain the startup time/date for the second-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 3	A726 to A728	These words contain the startup time/date for the third-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 4	A729 to A731	These words contain the startup time/date for the fourth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 5	A732 to A734	These words contain the startup time/date for the fifth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 6	A735 to A737	These words contain the startup time/date for the sixth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 7	A738 to A740	These words contain the startup time/date for the seventh-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 8	A741 to A743	These words contain the startup time/date for the eighth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 9	A744 to A746	These words contain the startup time/date for the ninth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.
Power ON Clock Data 10	A747 to A749	These words contain the startup time/date for the tenth-to-last time that power was turned ON. The data is BCD and the storage format is the same as words A720 to A722.

Power Interruption Time

The date and time that the PLC is turned OFF are stored in the following Auxiliary Area words. The number of power interruptions is also stored in the following Auxiliary Area word.

Name	Address	Description
Power Interruption Time	A512 and A513	Contain the time (in BCD) at which the power was interrupted. The contents are updated every time that the power is interrupted. A512.00 to A512.07: Seconds (00 to 59) A512.08 to A512.15: Minutes (00 to 59) A513.00 to A513.07: Hour (00 to 23) A513.08 to A513.15: Day of month (01 to 31)
Number of Power Interruptions	A514	Contains the number of times (in binary) that power has been interrupted since the power was first turned ON. To reset this value, overwrite the current value with 0000.

Total Power ON Time

The total amount of time that the PLC has been ON is stored in the following Auxiliary Area word.

Name	Address	Description
Total Power ON Time	A523	Contains the total time (in 16-bit binary) that the PLC has been ON in 10-hour increments. To reset this value, overwrite the current value with 0000 hex.

User Program and Parameter Revision Times

These Auxiliary Area words store the date and time that data was written to the user programs or parameters (i.e., PLC Setup, I/O tables, routing tables, or CPU Bus Unit Setups).

Name	Address	Description
User Program Date	A090 to A093	These words contain in BCD the date and time that the user program was last overwritten. A090.00 to A090.07: Seconds (00 to 59) A090.08 to A090.15: Minutes (00 to 59) A091.00 to A091.07: Hour (00 to 23) A091.08 to A091.15: Day of month (01 to 31) A092.00 to A092.07: Month (01 to 12) A092.08 to A092.15: Year (00 to 99) A093.00 to A093.07: Day of the week (00 to 06) 00: Sunday, 01: Monday, 02: Tuesday, 03: Wednesday, 04: Thursday, 05: Friday, 06: Saturday
Parameter Date	A094 to A097	These words contain in BCD the date and time that the parameters were last overwritten. A094.00 to A094.07: Seconds (00 to 59) A094.08 to A094.15: Minutes (00 to 59) A095.00 to A095.07: Hour (00 to 23) A095.08 to A095.15: Day of month (01 to 31) A096.00 to A096.07: Month (01 to 12) A096.08 to A096.15: Year (00 to 99) A097.00 to A097.07: Day of the week (00 to 06) 00: Sunday, 01: Monday, 02: Tuesday, 03: Wednesday, 04: Thursday, 05: Friday, 06: Saturday

Operation Start/End Times

These Auxiliary Area words automatically store the date and time that PLC operation was started and stopped.

Name	Address	Description
Operation Start Time	A515 to A517	The time that operation started as a result of changing the operating mode to RUN or MONITOR mode is stored here in BCD.*1 A515.00 to A515.07: Seconds (00 to 59) A515.08 to A515.15: Minutes (00 to 59) A516.00 to A516.07: Hour (00 to 23) A516.08 to A516.15: Day of month (01 to 31) A517.00 to A517.07: Month (01 to 12) A517.08 to A517.15: Year (00 to 99)
Operation End Time	A518 to A520	The time that operation stopped as a result of changing the operating mode to PROGRAM mode is stored here in BCD.*2 A518.00 to A518.07: Seconds (00 to 59) A518.08 to A518.15: Minutes (00 to 59) A519.00 to A519.07: Hour (00 to 23) A519.08 to A519.15: Day of month (01 to 31) A520.00 to A520.07: Month (01 to 12) A520.08 to A520.15: Year (00 to 99)

*1 The previous start time is stored after turning ON the power supply until operation is started.

*2 If an error occurs in operation, the time of the error will be stored. If the operating mode is then changed to PROGRAM mode, the time that PROGRAM mode was entered will be stored.

10-1-3 Free-running Timers

The system timers used after the power is turned ON are contained in Auxiliary Area words A0, A1, and A2. These timers can be used to calculate time intervals without using timer instructions.

Name	Address	Description
10-ms Incrementing Free Running Timer	A0	A value of 0000 hex is set when the power is turned ON and this value is automatically incremented by 1 every 10 ms. The value returns to 0000 hex after reaching FFFF hex (655,350 ms), and then continues to be automatically incremented by 1 every 10 ms.
100-ms Incrementing Free Running Timer	A1	A value of 0000 hex is set when the power is turned ON and this value is automatically incremented by 1 every 100 ms. The value returns to 0000 hex after reaching FFFF hex (6,553,500 ms), and then continues to be automatically incremented by 1 every 100 ms.
1-s Incrementing Free Running Timer	A2	A value of 0000 hex is set when the power is turned ON and this value is automatically incremented by 1 every second. The value returns to 0000 hex after reaching FFFF hex (65,535 s), and then continues to be automatically incremented by 1 every second.

Note When the operating mode is changed to RUN mode, automatic incrementing by 1 will continue.

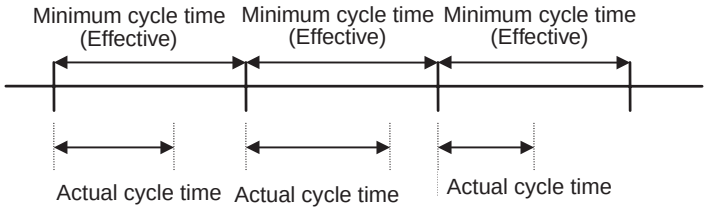
Example: The time interval between processing A and processing B can be calculated in increments of 10 ms by calculating the difference between the value in A0 for processing time A and the value in A0 for processing time B without using timer instructions.

10-2 Cycle Time/High-speed Processing

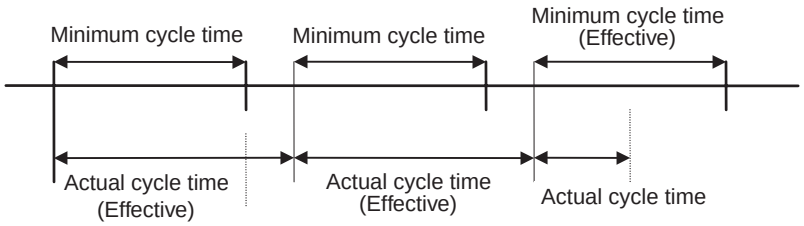
10-2-1 Minimum Cycle Time

A minimum (or fixed) cycle time can be set in PLCs. Variations in I/O response times can be eliminated by repeating the program with a fixed cycle time.

The minimum cycle time (0.1 to 32,000 ms) is specified in the PLC Setup in 0.1-ms units.

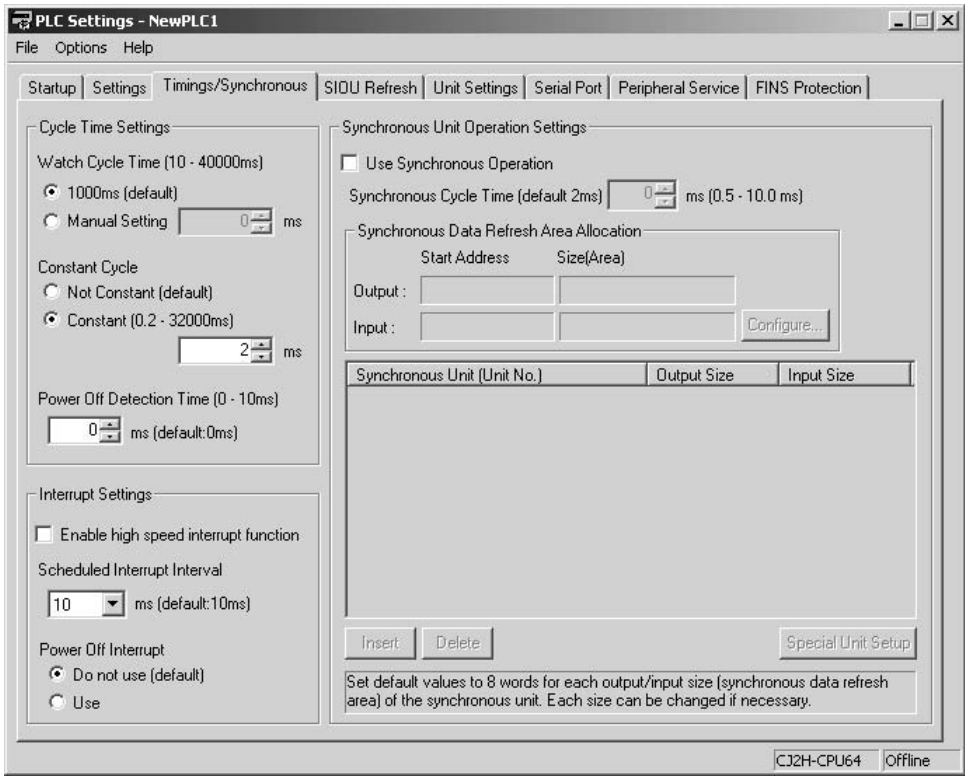


If the actual cycle time is longer than the minimum cycle time, the minimum cycle time function will be ineffective and the cycle time will vary from cycle to cycle.



● PLC Setup

When using the CX-Programmer, make the settings on the Timings/Synchronous Tab Page.



Additional Information

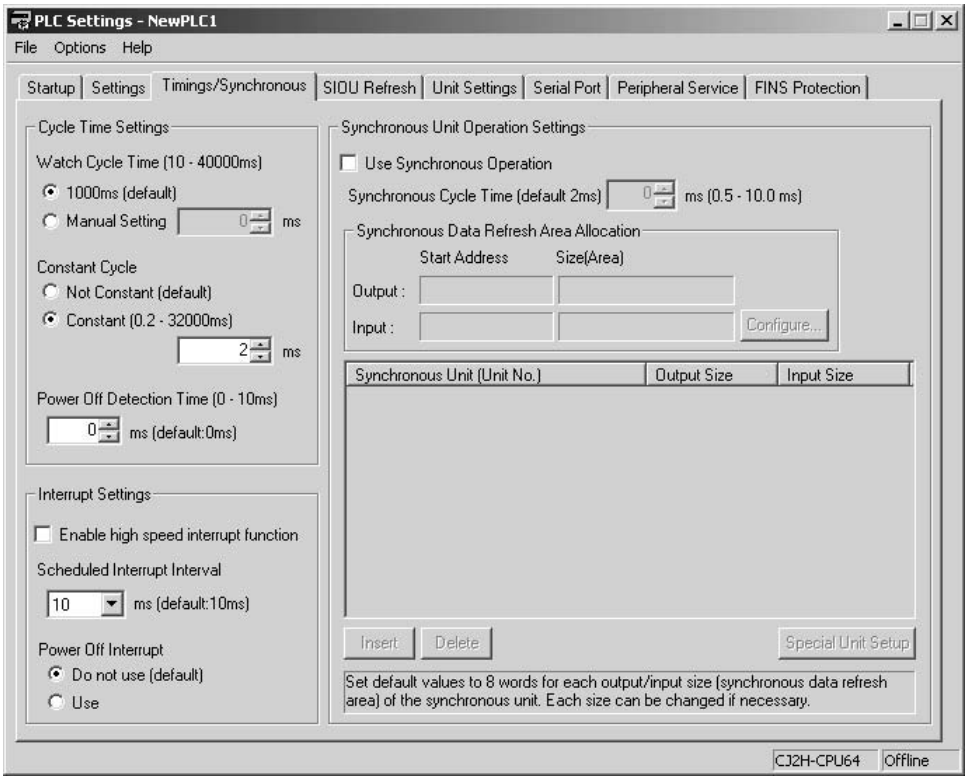
When the CPU Unit is operating in MONITOR mode, the minimum cycle time (constant cycle time) can be changed from the PLC Cycle Time Dialog Box of the CX-Programmer. (This function is supported only by CJ2H CPU Units with unit version 1.1 or later and CJ2M CPU Units.) For details, refer to *12-1 Monitoring the Cycle Time*.

10-2-2 Maximum Cycle Time

If the cycle time exceeds the maximum cycle time setting, the CPU Unit will stop operation. The Cycle Time Exceeded Flag (A401.08) will be turned ON. The default maximum cycle time is 1 s.

● PLC Setup

When using the CX-Programmer, set the maximum cycle time in the Watch Cycle Time Field on the Timings/Synchronous Tab Page.



● Auxiliary Area Flags and Words

Name	Address	Description
Cycle Time Exceeded Flag	A401.08	A401.08 will be turned ON if the cycle time exceeds the watch cycle time setting.

10-2-3 Monitoring the Cycle Time

Every cycle, the Auxiliary Area stores the maximum cycle time in A262 to A263 and the present cycle time in A264 to A265 and A266 to A267.

● Auxiliary Area Flags and Words

Name	Address	Description
Maximum Cycle Time (0.1-ms increments)	A262 and A263	The maximum cycle time in 0.1-ms increments is stored every cycle in 8-digit hexadecimal in the following range: 0 to 429,496,729.5 ms (0 to FFFF FFFF) The lower digits are stored in A262 and the upper digits are stored in A263.
Present Cycle Time (0.1-ms increments)	A264 and A265	The present cycle time in 0.1-ms increments is stored every cycle in 8-digit hexadecimal in the following range: 0 to 429,496,729.5 ms (0 to FFFF FFFF) The lower digits are stored in A264 and the upper digits are stored in A265.
Present Cycle Time (0.01-ms increments)	A266 and A267	The present cycle time value in 0.01-ms increments is stored every cycle in 8-digit hexadecimal in the following range: 0 to 42,949,672.95 ms (0 to FFFF FFFF) The lower digits are stored in A266 and the upper digits are stored in A267.

The CX-Programmer can be used to read the average cycle time for the last 8 cycles.

Additional Information

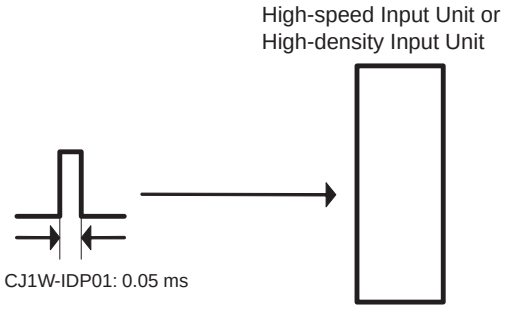
The following methods are effective ways to reduce the cycle time in CJ-series PLCs:

- Put tasks that aren't being executed in WAIT status.
- Use with JMP(004), CJP (510), or CJPN (511) together with JME(005) to jump program sections that do not need to be executed.

10-2-4 High-speed Inputs

When you want to receive pulses that are shorter than the cycle time, use the CJ1W-IDP01 High-speed Input Unit.

The high-speed inputs can receive pulses with a pulse width (ON time) of 0.05 ms for the CJ1W-IDP01 High-speed Input Unit.



Inputs that are input to internal memory are cleared when the inputs are refreshed.

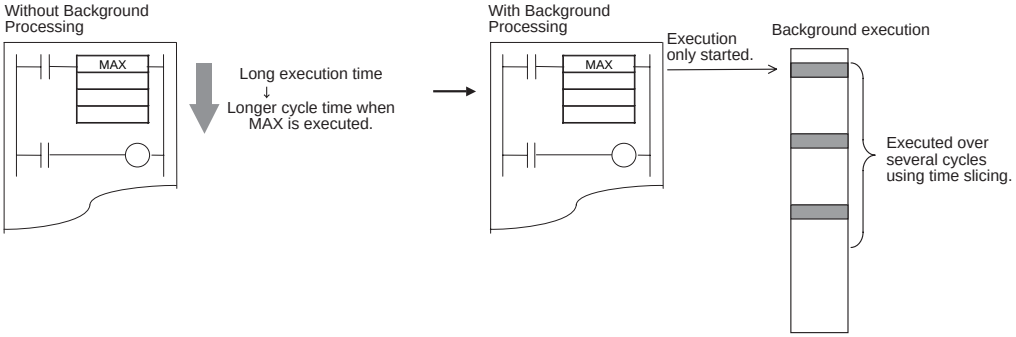
10-2-5 Background Execution

Table data processing (such as data searches) and text string processing (such as text string searches), require time to execute, and can create large fluctuations in the cycle time due to the extended amount of time required to execute them.

Background execution (time slicing) can be used to execute the following instructions over several cycles to help control fluctuations in the cycle time. The PLC Setup enables setting background execution for each type of instruction.

- Table data processing instructions
- Text string processing instructions
- Data shift instructions (ASYNCHRONOUS SHIFT REGISTER only)

Setting background execution for the above instructions can help control temporary increases in the cycle time.



Precautions for Correct Use

Background processing is not performed in interrupt tasks for CJ2H CPU Units when high-speed interrupts are enabled in the PLC Setup. An instruction processing error will occur.

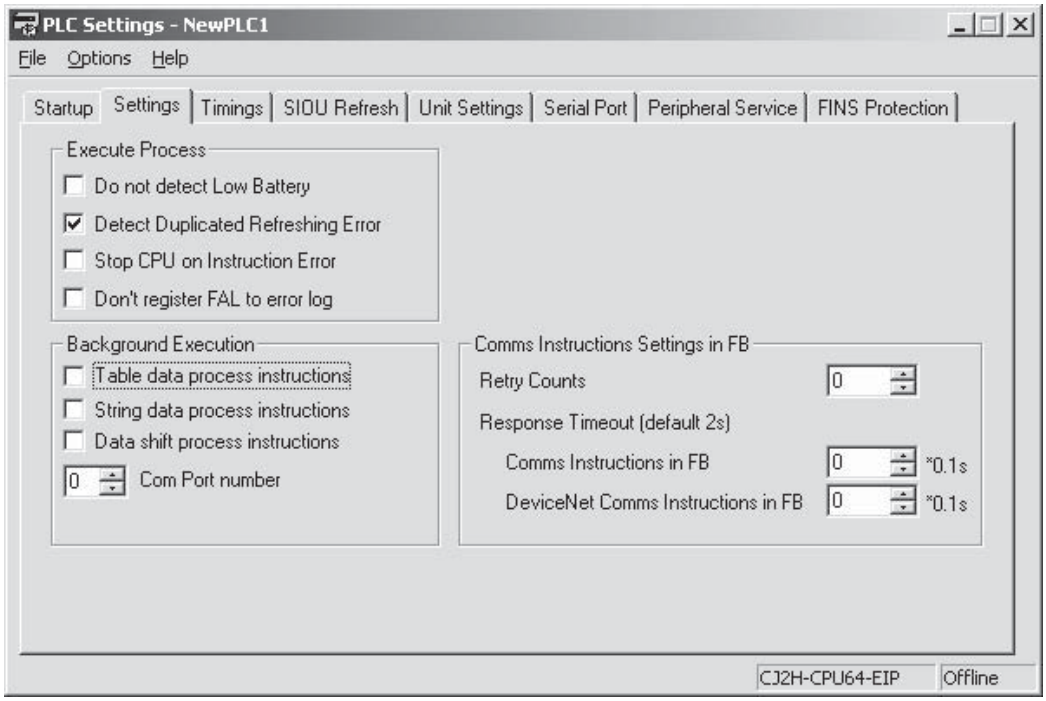
Applicable Instructions

Background processing will not be performed for the following instructions when they are used in function blocks. They will be executed using normal processing.

Group	Instruction	Mnemonic
Table Data Processing Instructions	DATA SEARCH	SRCH
	SWAP BYTES	SWAP
	Find Maximum Instructions	MAX
		MAXL
		MAXF
		MAXD
	Find Minimum Instructions	MIN
		MINL
		MINF
		MIND
	SUM	SUM
	FRAME CHECKSUM	FCS
Data Shift Instructions	ASYNCHRONOUS SHIFT REGISTER	ASFT
Text String Processing Instructions	MOVE STRING	MOV$
	CONCATENATE STRING	+$
	GET STRING LEFT	LEFT$
	GET STRING RIGHT	RGHT$
	GET STRING MIDDLE	MID$
	FIND IN STRING	FIND$
	STRING LENGTH	LEN$
	REPLACE IN STRING	RPLC$
	DELETE STRING	DEL$
	EXCHANGE STRING	XCHG$
	CLEAR STRING	CLR$
	INSERT INTO STRING	INS$

Procedure

1 To enable background execution for the required instructions, make the background execution settings on the **Settings** Tab Page in the PLC Settings Dialog Box from the CX-Programmer.



2 Set the logical port number to be used for background execution in the PLC Setup. This port number will be used for all instructions processed in the background.
One port is used for all background execution. Background execution for an instruction can thus not be started if background execution is already being performed for another instruction. Use the Communications Port Enabled Flag to control instructions specified for background execution so that no more than one instruction is executed at the same time.

3 If an instruction for which background execution has been specified is executed, execution will only be started in the cycle in which the execution condition was met and execution will not be completed in the same cycle.

4 When background execution is started, the Communications Port Enabled Flag for that port will be turned OFF.

5 Background execution will be continued over several cycles.

6 When processing has been completed, the Communications Port Enabled Flag for that port will be turned ON. This will enable another instruction to be executed in the background.

● Differences between Instructions Executed Normally and Instructions Executed in the Background

The differences between normal instruction execution and execution in the background are listed below.

Outputting to Index Registers (IR)

If MAX(182), MAXL(174), MAXF(176), MAXD(178), MIN(183), MINL(175), MINF(177), or MIND(179) is executed to output the I/O memory map address of the word containing the minimum or maximum value to an index register, the address will not be output to the index register and will be output to A595 and A596 instead.

To store the address in an index register in the same way as for normal execution, use a Data Move instruction (e.g., MOVL(498)) to copy the address in A595 and A596 to an index register.

Conditions Flags

Conditions Flags will not be updated following execution of instructions processed in the background. To access the Conditions Flag status, execute an instruction that affects the Conditions Flags in the same way, as shown in the following example, and then access the Conditions Flags.

Outputting to Index Register IR00

If SRCH(181) is executed to output the I/O memory map address of the word containing the matching value (the first word if there is more than one) to an index register, the address will not be output to the index register and will be output to A595 and A596 instead.

Outputting to Data Registers for SRCH(181)

If SRCH(181) is executed to output the matching data to a data register, the data will not be output to the data register and will be output to A597 instead.

Matching Text Strings

If SRCH(181) finds matching data, it will not turn ON the Equals Flag, but will turn ON A598.01 instead.

Instruction Errors

If an instruction execution error or illegal access error occurs for an instruction being processed in the background, the ER or AER Flags will not be turned ON and A395.10 will be turned ON instead. A395.10 will remain ON until the next time an instruction is processed in the background.

Outputting to Data Registers for MAX(182) or MIN(183)

If MAX(182), MAXL(174), MAXF(176), MAXD(178), MIN(183), MINL(175), MINF(177), or MIND(179) is executed with a data register (DR0 to DR15) specified as the output word for the minimum or maximum value, an instruction execution error will occur and the ER Flag will turn ON.

● Auxiliary Area Flags and Words

Name	Address	Description
Communications Port Enabled Flags	A202.00 to A202.07	Turns ON when a network instruction can be executed with the corresponding port number or background execution can be executed with the corresponding port number. Bits 00 to 07 correspond to communications ports 0 to 7. If the simple backup operation is used to perform a write or compare operation for a Memory Card, a communications port will be automatically allocated, and the corresponding flag will be turned ON during the operation and turned OFF when the operation has been completed.
Communications Port Error Flags	A219.00 to A219.07	Turns ON when an error occurred during execution of a network instruction. Turns OFF at normal completion. Bits 00 to 07 correspond to communications ports 0 to 7. If the simple backup operation is used to perform a write or compare operation for a Memory Card, a communications port will be automatically allocated. The corresponding flag will be turned ON if an error occurs and will be turned OFF if the simple backup operation ends normally.
Communications Port Completion Codes	A203 to A210	These words contain the completion codes for the corresponding port numbers when network instructions have been executed. The contents will be cleared to 0000 hex when background execution has been completed. Words A203 to A210 correspond to communications ports 0 to 7. If the simple backup operation is used to perform a write or compare operation for a Memory Card, a communications port will be automatically allocated, and a completion code will be stored in the corresponding word.
Background Execution ER/AER Flag	A395.10	Turns ON when an instruction execution error or illegal access error occurs in an instruction being executed in the background. Turns OFF when power is turned ON or operation is started.
Background Execution IR00 Output	A595 and A596	These words receive the output when the output of an instruction executed in the background is specified for an index register. No output will be made to IR00. Range: 0000 0000 to FFFF FFFF hex Lower 4 digits: A595, Upper 4 digits: A596
Background Execution DR00 Output	A597	This word receives the output when the output of an instruction executed in the background is specified for a data register. No output will be made to DR00. Range: 0000 to FFFF hex
Background Execution Equals Flag Output	A598.01	This flag is turned ON when matching data is found for a SRCH(181) executed in the background.

Additional Information

The internal logical ports in the CPU Unit are used both for background execution and the following instructions:

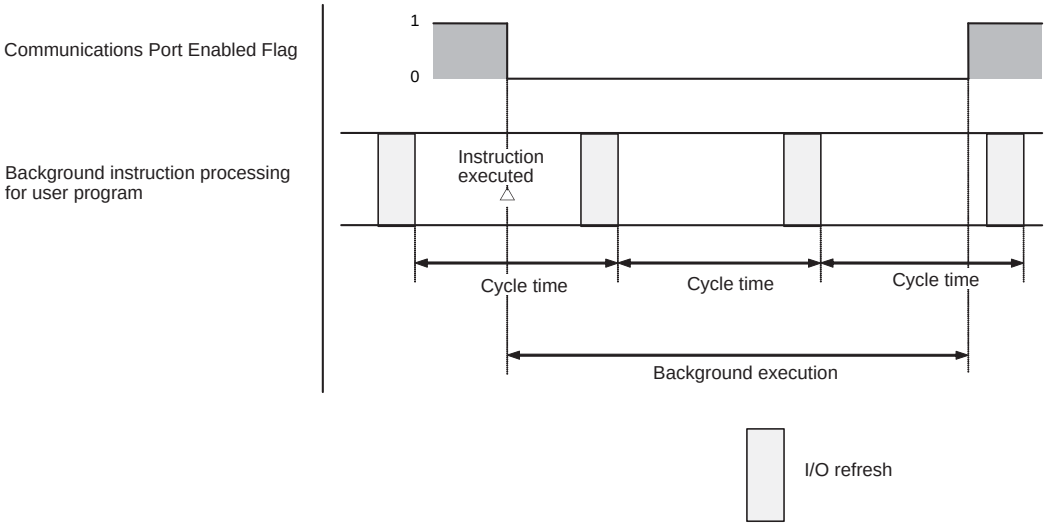
- SEND(090), RECV(098), and CMND(490) (Network Instructions)
- PMCR(260) (PROTOCOL MACRO)
- TXDU(256) and RXDU(255) (the no-protocol communications instructions used with Serial Communications Units)

Background instructions and the above instructions cannot be executed simultaneously on the same port. Use the Communications Port Enabled Flags to be sure that only one instruction is executed on each port at any one time.

Note If an instruction is specified for execution in the background for a port for which the Communications Port Enabled Flag is OFF, the ER Flag will turn ON and the background instruction will not be executed.

● Communications Port Enabled Flags

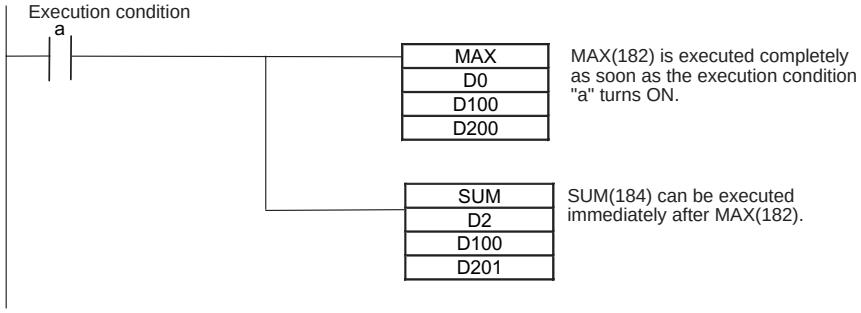
The Communications Port Enabled Flags are ON when the port is not being used and OFF when processing is being performed on the port.



Programming Example 1

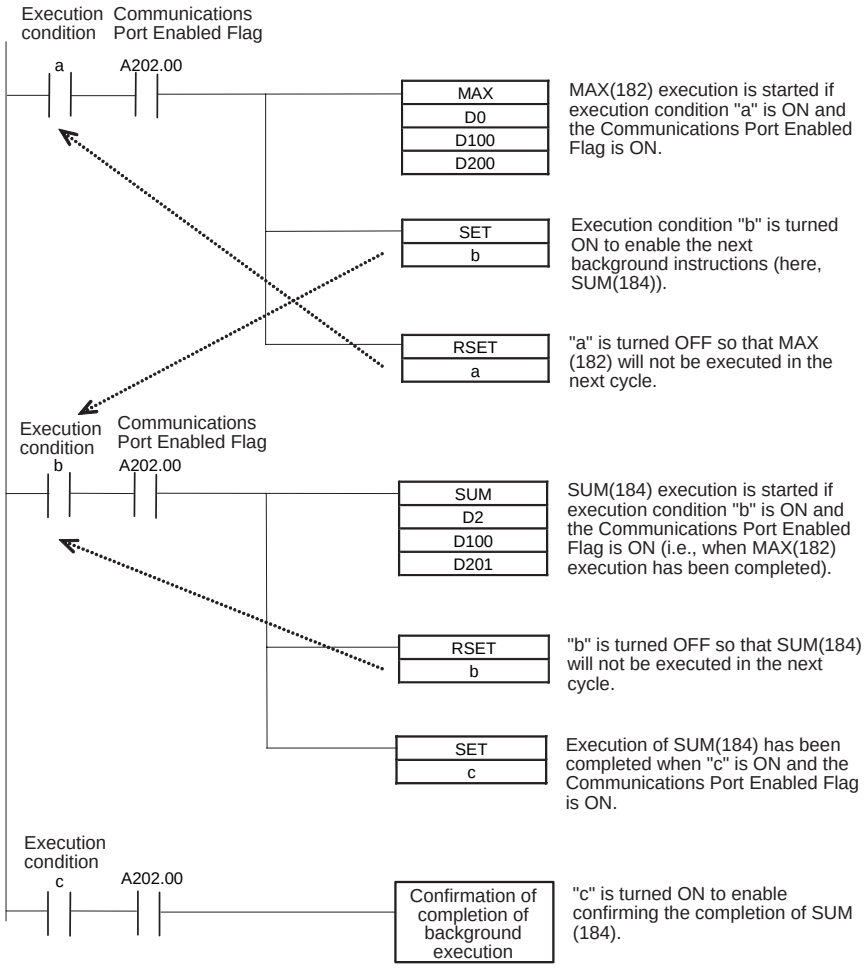
● Programming without Background Execution

As shown below, processing is completed when the instruction is executed.



● Programming with Background Execution

With background execution, the program is changed so that MAX(182) is executed only when the specified Communications Port Enabled Flag is ON (i.e., only when the port is not already being used for background execution or network communications). Also, input conditions are controlled with SET and RSET instructions to ensure that processing is performed in the correct order. (Communications port 0 is used for background execution in the following example.)

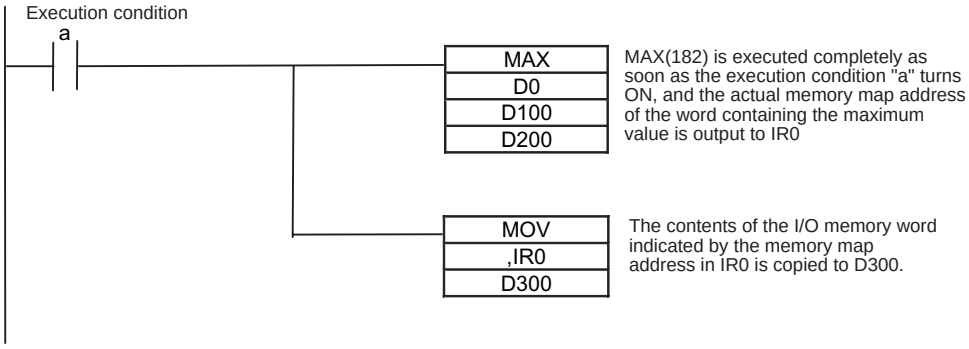


Programming Example 2

This example shows background execution when index register output is specified, as is possible for MAX(182), MIN(183), and SRCH(181).

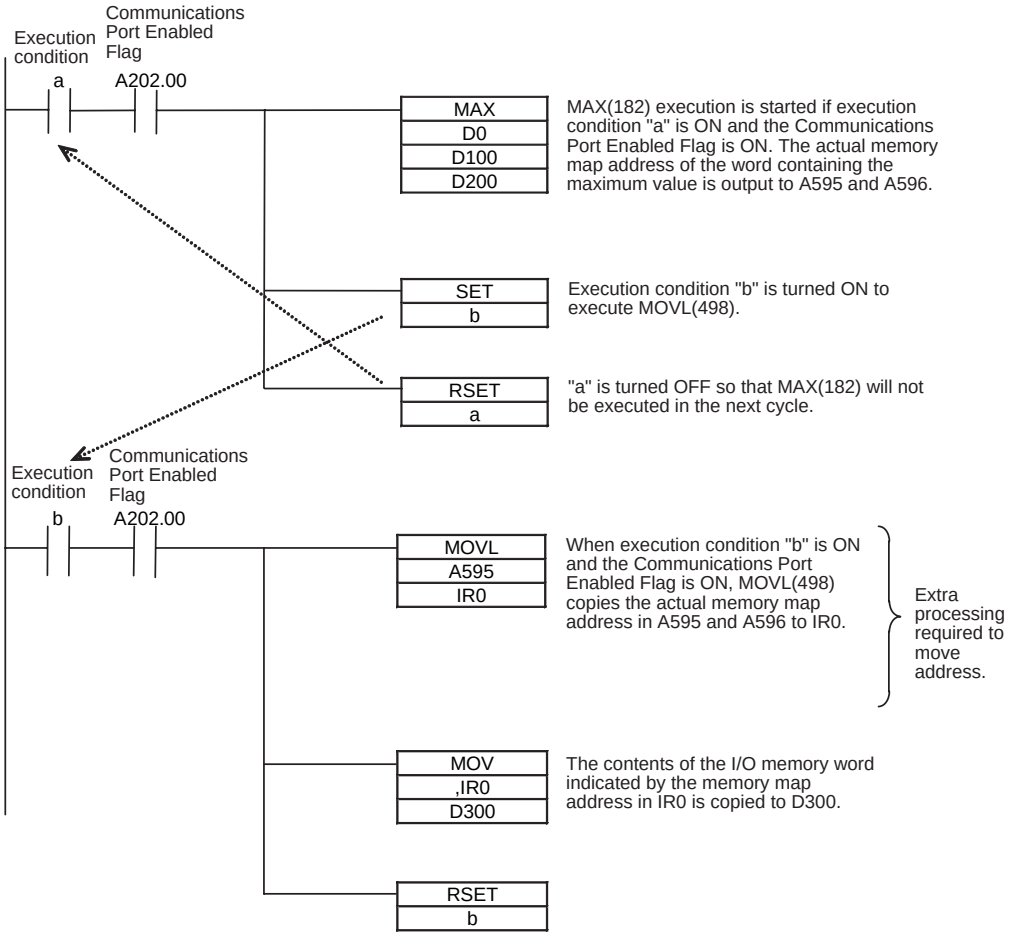
● Programming without Background Execution

As shown below, the PLC memory map address of the word containing the maximum or minimum value is output to an index register.



● Programming with Background Execution

With background execution, the PLC memory map address of the word containing the maximum or minimum value is output to A595 and A596. MOVL(498) is then used to move the PLC memory map address to the index register.

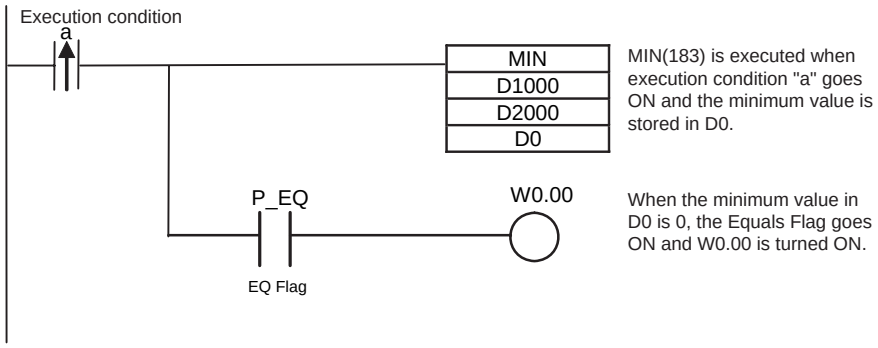


Programming Example 3

This example shows background execution when referencing Condition Flags.

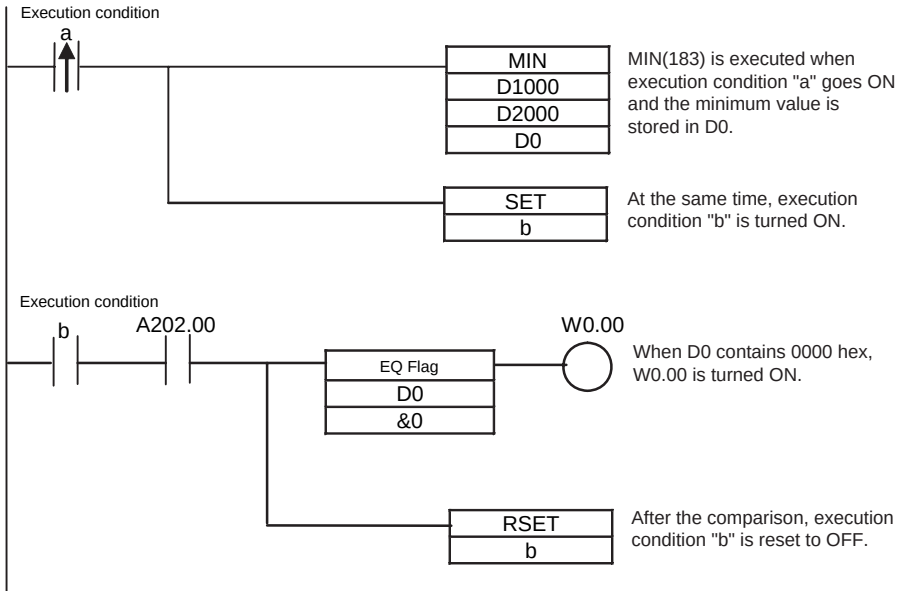
● Programming without Background Execution

To check whether the minimum value found by MIN(183) is 0, the status of the Equals Flag is checked just after execution of MIN(183).



● Programming with Background Execution

As shown in the following figure, an AND =(300) instruction is used to check whether the minimum value found by MIN(183) is 0.



10-2-6 High-speed Interrupt Function

High-speed interrupt function improves execution of interrupt tasks under certain restrictions (unit version 1.1 or later).

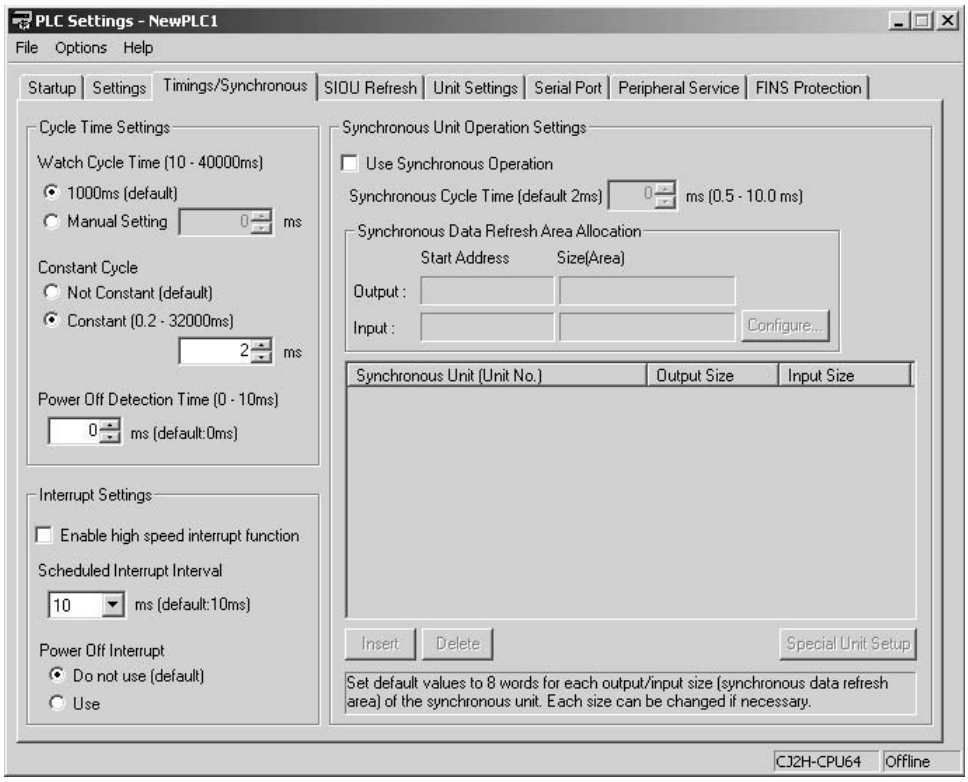
Shortening the Interrupt Overhead Time

Use the following procedures to shorten the overhead time (i.e., interrupt task startup time + Cyclic task return time) when executing I/O interrupt tasks, external interrupt tasks, or scheduled interrupt tasks.

Interrupt task type	Interrupt overhead time	
	High-speed interrupt function enabled	High-speed interrupt function disabled (default)
I/O interrupt tasks and external interrupt tasks	25 μs (Interrupt task startup time 17 μs + Cyclic task return time 8 μs)	37 μs (Interrupt task startup time 26 μs + Cyclic task return time 11 μs)
Scheduled interrupt task	21 μs (Interrupt task startup time 13 μs + Cyclic task return time 8 μs)	33 μs (Interrupt task startup time 22 μs + Cyclic task return time 11 μs)

● PLC Setup

When using the CX-Programmer, select the *Enable high-speed interrupt function* Check Box on the Timings/Synchronous Tab Page.



● Restrictions When High-speed Interrupt Function Is Enabled

The following restrictions apply when high-speed interrupt function is enabled.

- The following instructions (e.g., network communications instructions) cannot be used in interrupt tasks.

Mnemonic	Instruction name
EMBC(281)	SELECT EM BANK
SEND(090)*	NETWORK SEND
RECV(098)*	NETWORK RECEIVE
CMND(490)*	DELIVER COMMAND
PMCR(260)*	PROTOCOL MACRO
TXDU(256)	TRANSMIT VIA SERIAL COMMUNICATIONS UNIT
RXDU(255)	RECEIVE VIA SERIAL COMMUNICATIONS UNIT
EXPLT(720), EGATR(721), ESATR(722), ECHRD(723), and ECHWR(724)	EXPLICIT MESSAGE SEND (EXPLT(720)), EXPLICIT GET ATTRIBUTE (EGATR(721)), EXPLICIT SET ATTRIBUTE (ESATR(722)), EXPLICIT WORD READ (ECHRD(723), and EXPLICIT WORD WRITE (ECHWR(724))

* SEND2(491),RECV2(492), CMND2(493), and PMCR2(264) can be used in interrupt tasks when high-speed interrupt function is enabled.

- Instructions that are executed in the background (i.e., Table Data Processing Instructions, Text String Processing Instructions, and Data Shift Instructions) cannot be used in interrupt tasks.
- The data in the following Auxiliary Area words will not be valid: A440 (Maximum Interrupt Task Processing Time) and A441 (Interrupt Task with Maximum Processing Time).
- The performance of refreshing with the built-in EtherNet/IP port and EtherNet/IP Unit will decrease as shown in the following table when High-speed interrupt function is enabled.

	High-speed interrupt function enabled	Normal operation
Overhead processing time (additional time when using data links)	100 μs + Number of words transferred × 0.87 μs	100 μs + Number of words transferred × 0.33 μs
Execution time of DLNK(226) instruction for EtherNet/IP (immediate I/O refresh for CPU Bus Unit)	910 μs/1,000 words	430 μs/1,000 words

Minimum Time Interval for Scheduled Interrupt Tasks

A minimum time interval of 0.1 ms can be set for scheduled interrupt 0 (interrupt task 2) using MSKS(690). This is not possible for scheduled interrupt 1.

● Conditions for Setting the Minimum Time Interval for Scheduled Interrupt 0 to 0.1 ms

1 High-speed interrupt function must be enabled. (Select the *Enable high-speed interrupt function* Option in the PLC Setup.)

If High-speed interrupt function is disabled, specifying 0.1 ms in MSKS(690) will result in an instruction processing error.

2 Devices must not be directly connected to the peripheral port (USB) or serial port on the CPU Unit. A time interval of 0.1 ms may not be stable if Support Software (e.g., the CX-Programmer) or an NS-series Programmable Terminal is connected directly to the peripheral port (USB) or serial port on the CPU Unit. If a connection is required, make the connection using the built-in EtherNet/IP port (CJ2H-CPU6□-EIP only), or through the communications port of an EtherNet/IP Unit or a Serial Communications Unit.

Precautions for Correct Use

High-speed interrupt function improves execution of interrupt tasks under certain restrictions. Be sure to check the conditions for which use is possible before attempting operation. In particular, operation can be performed with the minimum time interval for scheduled interrupts set to 0.1 ms only on the condition that Support Software (e.g., the CX-Programmer) or an NS-series Programmable Terminal is not directly connected to the peripheral port (USB) or serial port on the CPU Unit. If a direction connection is made, operation may not be performed at a time interval of 0.1 ms. If Support Software must be connected directly to the CPU Unit for maintenance of other reasons, confirm that equipment will not be affected if the schedule interrupt is not executed for a time interval of 0.1 ms before making the connection.

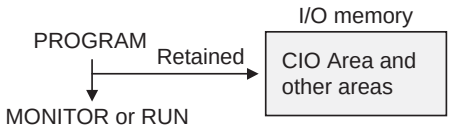
10-3 Startup Settings and Maintenance

10-3-1 Holding Settings for Operating Mode Changes and at Startup

Operating Mode Changes

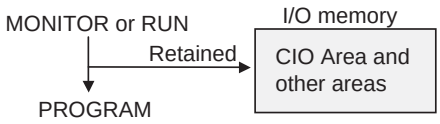
● Starting Program Execution

Turn ON the IOM Hold Bit (A500.12) to retain all data in I/O memory when the CPU Unit is switched from PROGRAM mode to RUN/MONITOR mode to start program execution.



● Stopping Program Execution

When the IOM Hold Bit (A500.12) is ON, all data in I/O memory will also be retained when the CPU Unit is switched from RUN/MONITOR mode to PROGRAM mode to stop program execution.



The following areas are held for the IOM Hold Bit: CIO Area (I/O Area, Data Link Area, CPU Bus Unit Area, Special I/O Unit Area, DeviceNet Area, and Internal I/O Area), Work Area, Timer Completion Flags, and Timer PVs.

● Auxiliary Area Flags and Words

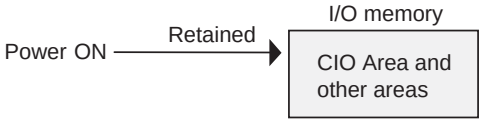
Name	Address	Description
IOM Hold Bit	A500.12	When this bit is ON, all of the I/O memory will be retained when the operating mode is changed between PROGRAM and RUN/MONITOR mode or the power is turned ON. ON: I/O memory will be retained when the operating mode is changed. OFF: I/O memory will be cleared when the operating mode is changed.

When the IOM Hold Bit is ON, all outputs from Output Units will be maintained when program execution stops. When the program starts again, outputs will have the same status that they had before the program was stopped.

(When the IOM Hold Bit is OFF, instructions will be executed after the outputs have been cleared.)

PLC Power ON

In order for all data in I/O memory to be retained when the PLC is turned ON, the IOM Hold Bit (A500.12) must be ON and it must be protected in the PLC Setup.

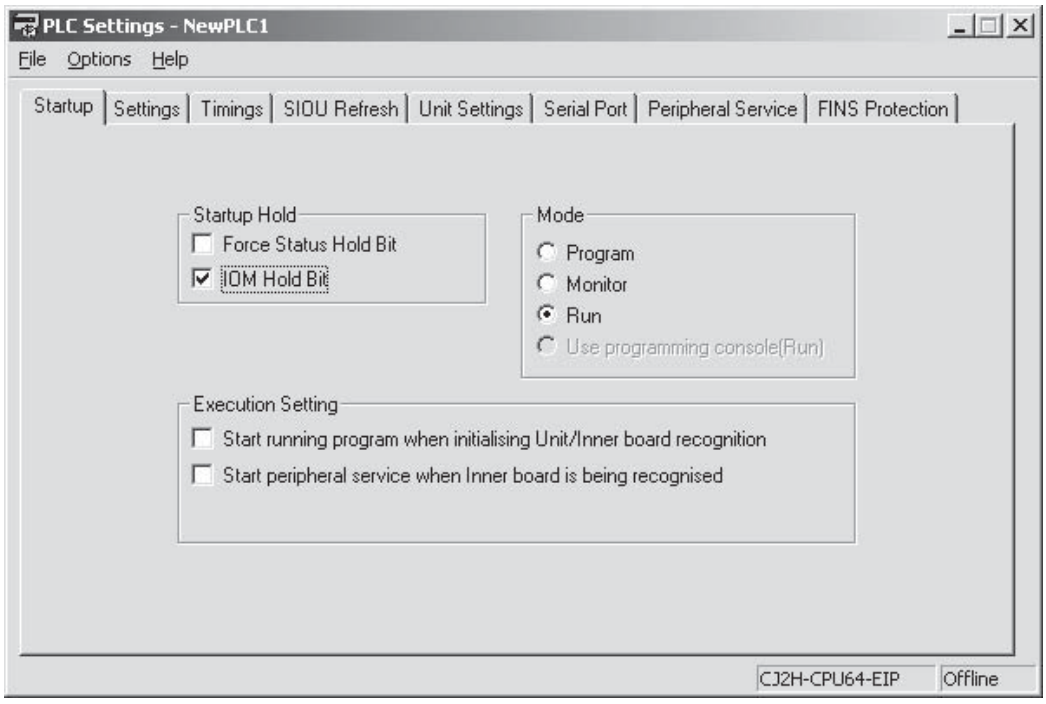


● Auxiliary Area Flags and Words

Name	Address	Description
IOM Hold Bit	A500.12	When this bit is ON, all of I/O memory will be retained when the operating mode is changed between PROGRAM and RUN/MONITOR or the power is turned ON. ON: I/O memory will be retained when the operating mode is changed. OFF: I/O memory will be cleared when the operating mode is changed.

● PLC Setup

When using the CX-Programmer, select the *IOM Hold Bit* Check Box in the Startup Hold Area on the Startup Tab Page to make the setting.

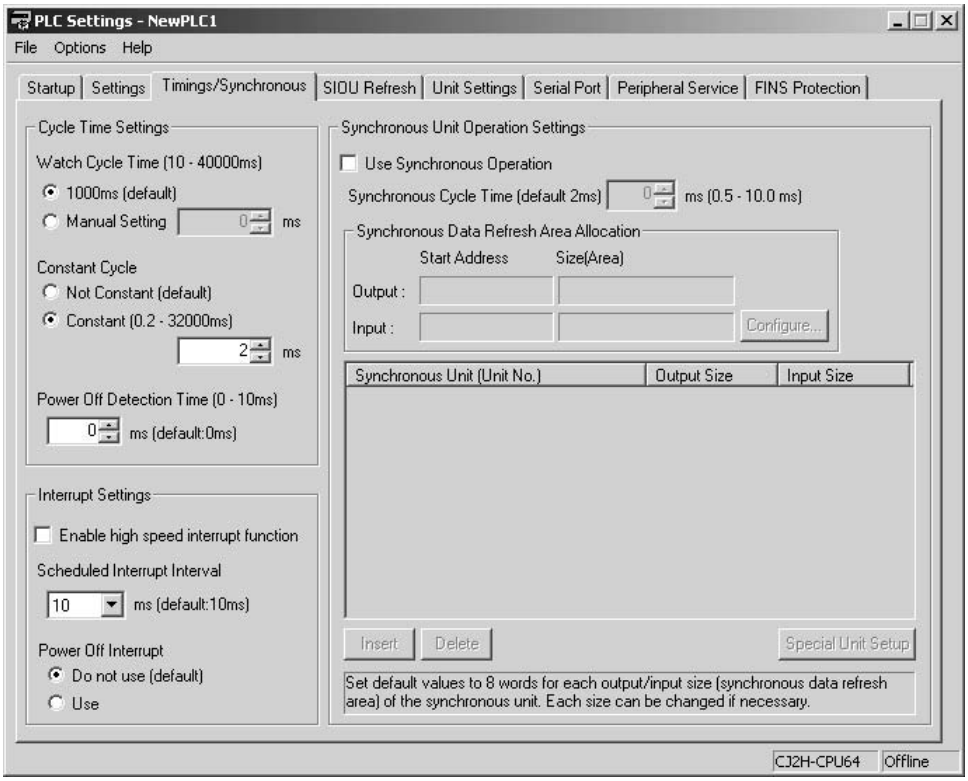


10-3-2 Power OFF Detection Delay Setting

By default, an AC power interruption of 10 ms or longer (2 ms for a DC power interruption) will be detected about 10 to 25 ms (2 to 5 ms for DC power supplies) after the power supply voltage drops below 85% of the minimum rated value (80% for DC power supplies). There is a setting in the PLC Setup that can extend this time. When the power OFF interrupt task is enabled, it will be executed when the power interruption is confirmed, otherwise the CPU Unit will be reset and operation will be stopped.

PLC Setup

When using the CX-Programmer, make the setting in the *Power Off detection time* Field on the Timings/Synchronous Tab Page.



Additional Information

If you will not use the power OFF interrupt task, set the power OFF detection time to 10 ms or less as a measure for power interruptions. If the CJ1W-PD022 Power Supply Unit is used, however, a delay cannot be set, so use the default setting of 0 ms.

10-3-3 Disabling Power OFF Interrupts

Areas of the program can be protected from power OFF interrupts so that they will be executed before the CPU Unit is reset even if the power supply is interrupted. This is achieved by using the DISABLE INTERRUPTS (DI(693)) and ENABLE INTERRUPTS (EI(694)) instructions.

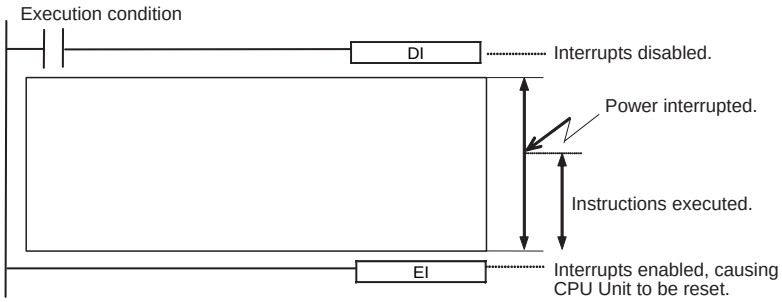
This function can be used with sets of instructions that must be executed as a group, e.g., so that execution does not start with intermediate stored data the next time power is turned ON.

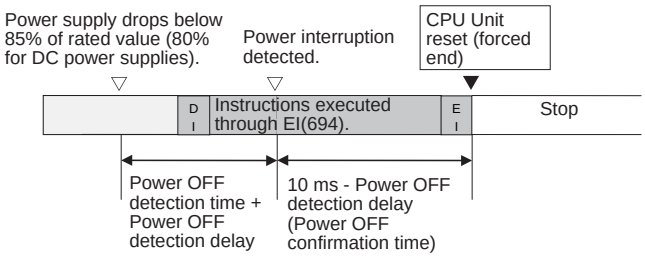
Procedure

1 Set the Disable Setting for Power OFF Interrupts in A530 to A5A5 hex to enable disabling Power OFF Interrupts.

2 Enable disabling Power OFF Interrupts in the PLC Setup (this is the default setting).

3 Use DI(693) to disable interrupts before the program section to be protected and then use EI(694) to enable interrupts after the section. All instructions between DI(693) and EI(694) will be completed before the Power OFF Interrupt is executed even if the power interruption occurs while executing the instructions between DI(693) and EI(694).





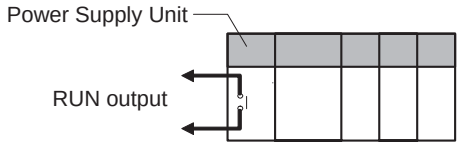
● Auxiliary Area Flags and Words

Name	Address	Meaning
Disable Setting for Power OFF Interrupts	A530	Enables using DI(693) to disable power OFF interrupt processing (except for execution of the Power OFF Interrupt Task) until EI(694) is executed. A5A5 hex: Enables using DI(693) to disable power OFF interrupt processing Any other value: Disables using DI(693) to disable power OFF interrupt processing

10-3-4 RUN Output

RUN Output

The CJ1W-PA205R Power Supply Unit is equipped with a RUN output. This output point is ON (closed) when the CPU Unit is operating in RUN or MONITOR mode.



Name	Operation
RUN output	MONITOR or RUN mode: ON (closed) PROGRAM mode: OFF (open)

This RUN output can be used to create an external safety circuits, such as an emergency stop circuit that prevents an Output Unit's external power supply from providing power unless the PLC is ON.

Note When a Power Supply Unit without a RUN output is used, an equivalent output can be created by programming the Always ON Flag (A1) as the execution condition for an output point from an Output Unit.

Precautions for Safe Use

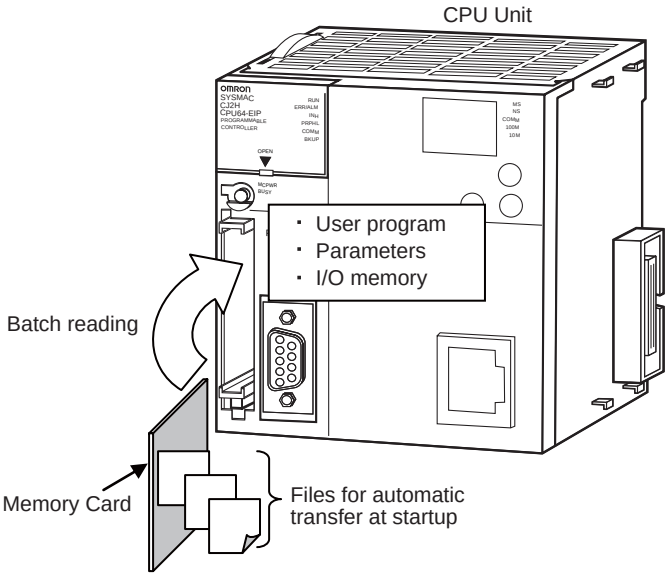
If Output Unit's external power supply goes ON before the PLC's power supply, the Output Unit may malfunction momentarily when the PLC first goes ON. To prevent any malfunction, add an external circuit that prevents the Output Unit's external power supply from going ON before the power supply to the PLC itself. Create a fail-safe circuit like the one described above to ensure that power is supplied by an external power supply only when the PLC is operating in RUN or MONITOR mode.

10-3-5 Automatic Transfer at Startup

Overview

Automatic transfer at startup is used to read the user program, parameters, and I/O memory data from a Memory Card to the CPU Unit when the power is turned ON.

The files for automatic transfer at startup can be created in the Memory Card Window of the CX-Programmer and stored in the Memory Card.



This function cannot be used to read EM file memory.

Additional Information

I/O memory will be unstable if operation is performed without the battery, so this function can be used to ensure that the correct values are always used.

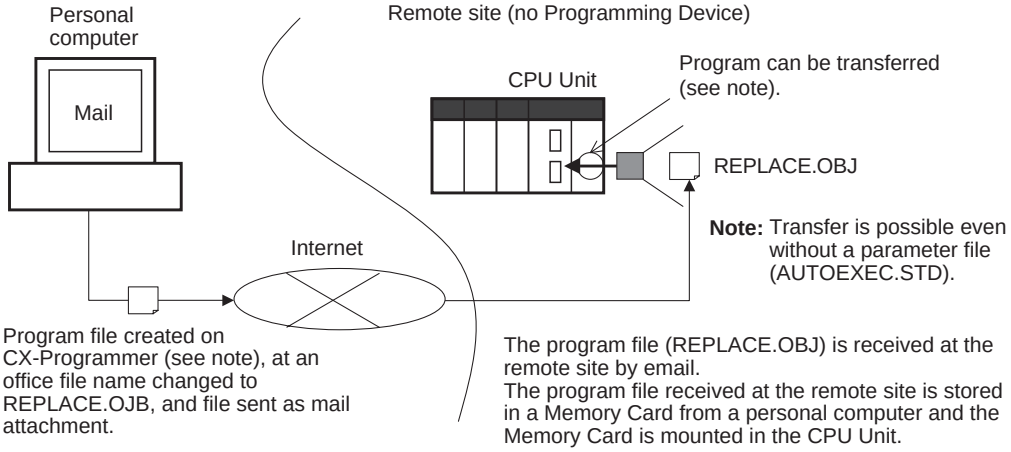
Types of Automatic Transfer at Startup

There are two ways to perform automatic transfer at startup: transferring either with or without a parameter file (AUTOEXEC.STD). The file name to be created depends on which of the two transfer types is used.

Type of automatic transfer at startup	File name	Application
Transfer with parameter file	AUTOEXEC or ATEXEC□□	Overwriting the program and network settings
Transfer without parameter file	REPLACE or REPLC□□	Overwriting the program

● Example Application for Automatically Transferring Files without a Parameter File

A program/network symbol file (.OBJ) can be created offline in an office (i.e., without the actual devices) and transferred to a remote location without a parameter file (.STD). The program/network symbol file can be stored in a Memory Card at the remote site without using a Programming Device and the Memory Card can be used to automatically transfer the program to the CPU Unit at startup.

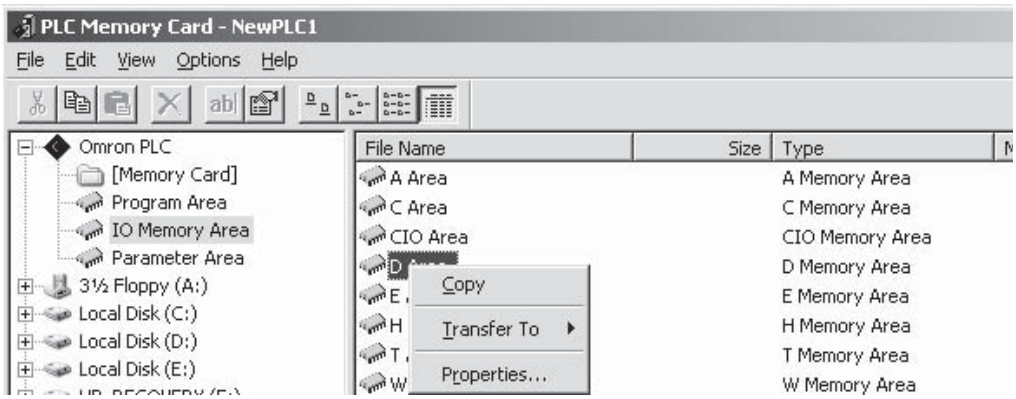


Procedure

● Creating and Transferring Files for Automatic Transfer at Startup

Create the file for automatic transfer at startup by using the CX-Programmer and transfer it to the Memory Card mounted in the CPU Unit.

- Program Area and Parameter Area:
  Right-click and select ***Transfer to*** – ***Memory Card***. Input the name of the file for automatic transfer at startup, and then create and transfer it.
- I/O Memory Area:
  Right-click the D Area or E Area in the pane on the right. Input the name of the file for automatic transfer at startup, and then create and transfer it.

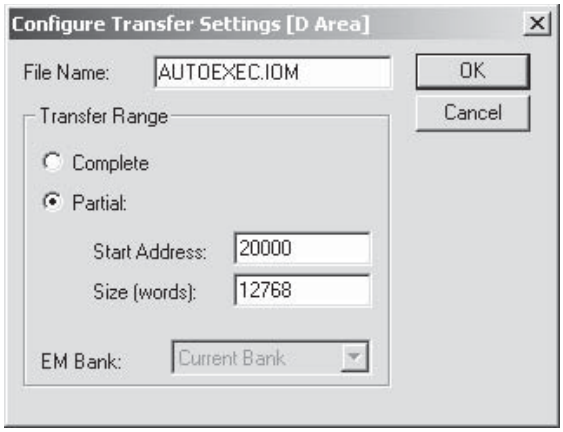


The following file names are used for automatic transfer at startup.

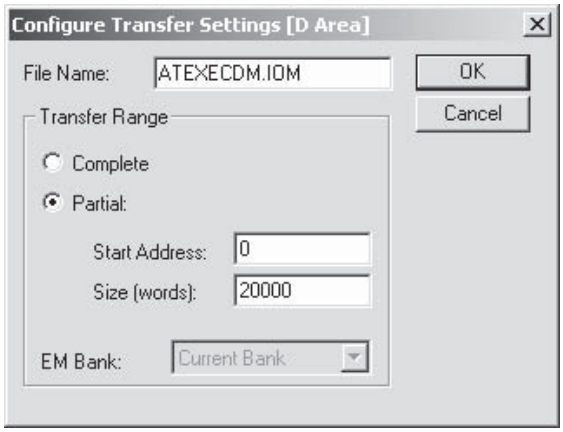
File type	Contents	Automatic transfer at startup with parameter file	Required?	Automatic transfer at startup without parameter file	Required?
Program/network symbol file	User program	AUTOEXEC.OBJ	Yes	REPLACE.OBJ	Yes
Parameter file	• PLC name • PLC Setup • Registered I/O tables • Routing tables • CPU Bus Unit Setup Area	AUTOEXEC.STD	Yes	None	---
Data files	Allocated DM Area words for Special I/O Units and CPU Bus Units (D20000 to D32767)	AUTOEXEC.IOM	---	REPLACE.IOM	---
	DM Area (D0 to D19999)	ATEXECDM.IOM	---	REPLACDM.IOM	---
	EM Area	AUTOEM□□.IOM (□□: 00 to 18 hex)	---	RPLCEM□□.IOM (□□:00 to 18 hex)	---

For example, specify the following start address and size for the DM Area.

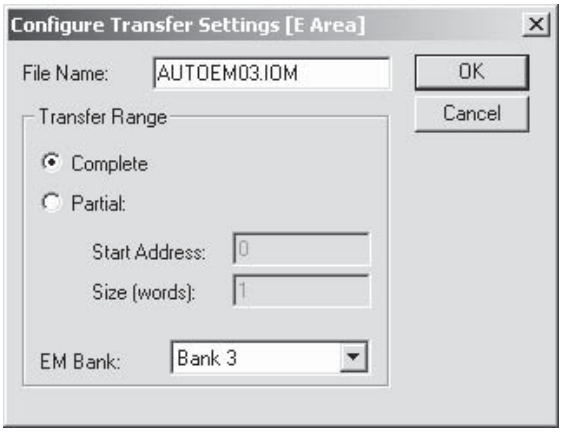
- Allocated DM Area Words for Special I/O Units and CPU Bus Units



- DM Area:



- EM Area:



● Performing Automatic Transfer at Startup

1 Turn OFF the PLC.

2 Turn ON pin 2 on the DIP switch on the front of the CPU Unit.

The simple backup will operate if pin 7 is ON. Be sure it is set to OFF.

3 Mount the Memory Card in the CPU Unit with the required files already stored.

4 Turn ON the PLC.

Automatic transfer at startup will start, and the BUSY indicator will flash yellow. The BUSY indicator will turn OFF when transfer has been completed. If automatic transfer at startup fails, a card transfer error will occur, and CPU Unit operation will stop.

● DIP Switch on Front of CPU Unit

Switch pin number	Name	Setting
2	Automatic transfer at startup	ON: Perform automatic transfer at startup OFF: Do not perform automatic transfer at startup

● Auxiliary Area Flags and Words

Name	Address	Description
Card Transfer Error Flag (fatal error)	A401.03	Turns ON when automatic transfer at startup from the Memory Card fails. CPU Unit operation will stop, and the ERR/ALM indicator on the front of the CPU Unit will light.

Data Files for Automatic Transfer at Startup

The following six types of data files are used for automatic transfer at startup.

Data files for automatic transfer at startup		Contents	Range	Start address	Size
Automatic transfer at startup with parameter file	Automatic transfer at startup without parameter file				
AUTOEXEC.IOM	REPLACE.IOM	Allocated DM Area words for Special I/O Units and CPU Bus Units (D20000 to D32767)	Partial	20000	12,768 words
ATEXECDM.IOM	REPLACDM.IOM	DM Area (D00000 to D19999)	Entire area	0	20,000 words
AUTOEM□□.IOM (□□: 00 to 18)	AUTOEM□□.IOM (□□: 00 to 18)	EM	Entire area	---	---

Be sure to set each of the above area types and the start addresses first when creating a data file for automatic transfer at startup. For the storage size, set the size until the last address of the area type.

At startup, all the data on the data files in the Memory Card will be transferred starting at D20000, D0, and E□_0. If there are not enough words for the entire area, the remaining words in the area will not be changed.

Automatic Transfer at Startup with Parameter File

Use the following file names. The files listed as being required in the right column must be on the Memory Card to perform automatic transfer at startup.

File type	File name	Extension	Contents	Description	Required?
Program/network symbol file	AUTOEXEC	.OBJ	User program and network symbols*1	• Programs in cyclic tasks and interrupt tasks. • This file must be on the Memory Card to perform automatic transfer at startup. • This parameter files (AUTOEXEC.STD) must also be on the Memory Card to perform automatic transfer at startup.	Yes
Parameter file	AUTOEXEC	.STD	• PLC name • PLC Setup • I/O tables • Routing tables • CPU Bus Unit Setup Area	• Parameter area data for the CPU Unit • When the power is turned ON, the parameters will be automatically stored in a specified location in the CPU Unit. • The user does not need to specify the data individually in the file. • The parameter file must be on the Memory Card to perform automatic transfer at startup when the program/network symbol file is AUTOEXEC.OBJ.	Yes
Data files	AUTOEXEC	.IOM	DM Area data (Contains the specified number of words of data starting at D20000.)	• Store the DM Area data starting from D20000 in the file named AUTOEXEC.IOM. • At startup, all data from the start of the file will be transferred staring at D20000. If there are not enough words for the entire area, the remaining words in the area will not be changed. • This file is not necessary to perform automatic transfer at startup.	---
	ATEXECDM	.IOM	DM Area data (Contains the specified number of words of data starting at D0.)	• Store the DM Area data starting at D0 in the file named ATEXECDM.IOM. • At startup, all data from the start of the file will be transferred starting at D0. If there are not enough words for the entire area, the remaining words in the area will not be changed. • This file is not necessary to perform automatic transfer at startup.*2	---

*1 Network symbols are supported only for the CJ2H-CPU6□-EIP and CJ2M-CPU3□. If a symbols file (.OBJ) is transferred to the CJ2H-CPU6□ or CJ2M-CPU3□, A401.03 (Card Transfer Error Flag) will turn ON.

*2 The ATEXECDM.IOM file will be given priority if its contents overlaps with the contents of the AUTOEXEC.IOM file.

File type	File name	Extension	Contents	Description	Required?
Data files, continued	AUTOEM□□	.IOM	EM Area data with bank No.□□ (Contains the specified number of words of data starting at E□□_0.)	• Store the EM Area data for bank No. □ starting at E□_0 in the file named AUTOEM□□.IOM. (The box refers to bank No. 0 to C. The maximum bank number of banks depends on the model of CPU Unit.) • At startup, all data from the start of the file will be transferred starting at E□_0. If there are not enough words for the entire area, the remaining words in the area will not be changed. • This file is not necessary to perform automatic transfer at startup.*2	---

If the contents of the AUTOEXEC.IOM and ATEXECDM.IOM files overlap, the ATEXECDM.IOM file will be given priority in writing. (This is because the files are loaded in the following order: AUTOEXEC.IOM and then ATEXECDM.IOM.)

Additional Information

Automatic transfer at startup and replacing the entire program using the Auxiliary Area control bits can be used in combination. It is possible to replace the program with another program using automatic transfer at startup, and then use an Auxiliary Area control bit to replace the program again during operation.

Automatic Transfer at Startup without Parameter File

Use the following file names. The files listed as being required in the right column must be on the Memory Card to perform automatic transfer at startup.

File type	File name	Extension	Contents	Description	Required?
Program/network symbol file	REPLACE	.OBJ	User program	• Contents is the same as AUTOEXEC.OBJ. • The program/network symbol file will be transferred at startup even if the parameter file (AUTOEXEC.STD) is not on the Memory Card.	Yes
Parameter file	Not required	---	---	• The parameter file will not be transferred no matter what file name is used in the Memory Card.	---
Data files	REPLACE	.IOM	DM Area data (Contains the specified number of words of data starting at D20000.)	• Contents is the same as with AUTOEXEC.IOM. • The data in this file will be transferred at startup if the program/network symbol file (REPLACE.OBJ) is on the Memory Card.	---
	REPLCDM	.IOM	DM Area data (Contains the specified number of words of data starting at D0.	• Contents is the same as with ATEXECDM.IOM. • The data in this file will be transferred at startup if the program/network symbol file (REPLACE.OBJ) is on the Memory Card.	---
	RPLCEM□□	.IOM	EM Area data with bank No. □□ (Contains the specified number of words of data starting at E□□_0.)	• Contents is the same as with AUTOEM□□.IOM. • The data in this file will be transferred at startup if the program/network symbol file (REPLACE.OBJ) is on the Memory Card.	---



Additional Information

Setup data for the Special I/O Units and CPU Bus Units can be read from the Memory Card at startup by storing data from the allocated DM Area words for Special I/O Units (D20000 to D29599) and allocated DM Area words for CPU Bus Units (D30000 to D31599) on the Memory Card as an AUTOEXEC.IOM file. Using Memory Cards in this way enables managing a library of system data for Special I/O Units and CPU Bus Units for each piece of equipment.

Approximate Times Required for Automatic Transfer at Startup

Size of user program and network symbols	Time required for automatic transfer (from turning ON power to start of operation)
User program: 120K steps	9 s
User program: 120K steps, Network symbols: 20,000	59 s

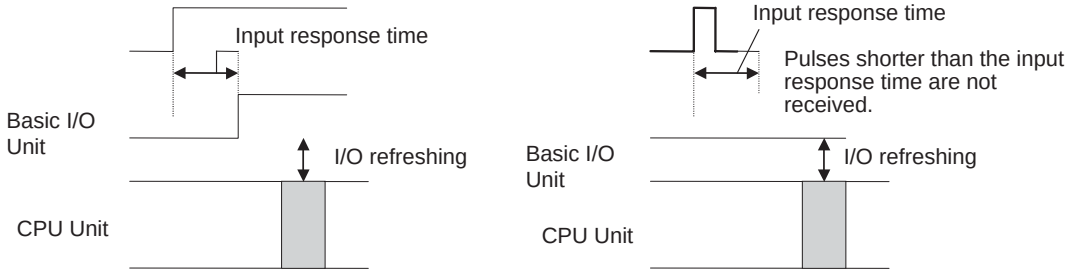
10-4 Unit Management Functions

10-4-1 Basic I/O Unit Management

● Setting the Input Response Time for Basic I/O Units

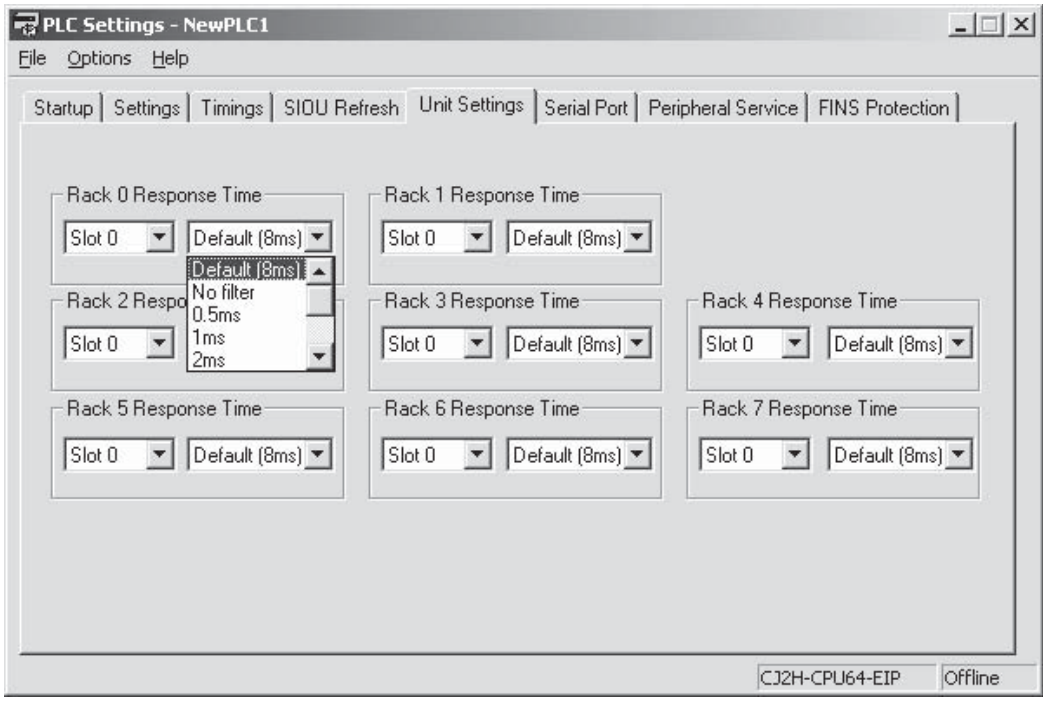
The input response times for Basic I/O Units can be set by Rack and Slot number. Increasing the input response time reduces the effects of chattering and noise. Decreasing the input response time (but keeping the pulse width longer than the cycle time) allows reception of shorter input pulses.

Note Pulses shorter than the cycle time can be input with a High-speed Input Unit. Refer to *10-2-4 High-speed Inputs* for details.



PLC Setup

When using the CX-Programmer, make the settings in the areas for specifying the response time for racks 0 to 7 on the Unit Settings Tab Page.



Auxiliary Area Flags and Words

Name	Address	Description
I/O Response Times in Basic I/O Units	A220.00 to A259.15	Contains the current I/O response times for CJ-series Basic I/O Units.

Basic I/O Unit Error Management

The following table gives the error information related to Basic I/O Units.

Auxiliary Area Flags and Words

Name	Address	Description
Basic I/O Unit Error Flag (Non-fatal error)	A402.12	ON when an error has occurred in a Basic I/O Unit.
Basic I/O Unit Error, Rack Number	A408.08 to A408.15	Contains the binary rack number where the error occurred (when A402.12 is ON) when an error has occurred in a Basic I/O Unit.
Basic I/O Unit Error, Slot Number	A408.00 to A408.07	Contains the binary slot number where the error occurred (when A402.12 is ON) when an error has occurred in a Basic I/O Unit.
Basic I/O Unit Status Area (rack 0 slot 0 to rack 3 slot 9)	A050.00 to A069.15	Indicates alarm status (load short-circuit protection) for Basic I/O Units.
Interrupt Input Unit Position Error Flag	A405.08	ON when the Interrupt Input Unit is not connected in one of the four positions (slots 0 to 3) next to the CPU Unit on the CPU Rack.

10-4-2 CPU Bus Unit Flags/Bits

● Restarting and Initializing CPU Bus Units

With CPU Bus Units, changes to initial settings for allocated DM Area words for the CPU Unit can be enabled by turning ON the following Restart Bits without turning OFF the PLC. Normally, this operation is performed from the user program.

Auxiliary Area Flags and Words

Name	Address	Description
CPU Bus Unit Restart Bits	A501.00 to A501.15	These bits correspond to CPU Bus Units 00 to 15 (unit numbers 0 to F). Turn ON a bit to restart the corresponding Unit.
CPU Bus Unit Initialization Flags	A302.00 to A302.15	These flags correspond to CPU Bus Units 00 to 15 (unit numbers 0 to F). A flag will be ON while the corresponding Unit is initializing after the power is turned ON or the Unit's Restart Bit (in A501.00 to A501.15) is turned ON.

● CPU Bus Unit Errors

The following table gives error information related to CPU Bus Units.

Auxiliary Area Flags and Words

Name	Address	Description
CPU Bus Unit Number Duplication Flags	A410.00 to A410.15	The Duplication Error Flag (A401.13) and the corresponding flag in A410 will be turned ON when a CPU Bus Unit's unit number has been duplicated. Bits 00 to 15 correspond to unit numbers 0 to F.
CPU Bus Unit Setting Error Flag (Non-fatal error)	A402.03	ON when an installed CPU Bus Unit does not match the CPU Bus Unit registered in the I/O table.
CPU Bus Unit Setting Error, Unit Number Flags	A427.00 to A427.15	When a CPU Bus Unit Setting Error occurs, A402.03 and the corresponding flag in A427 are turned ON. Bits 00 to 15 correspond to unit numbers 0 to F.
CPU Bus Unit Error Flag (Non-fatal error)	A402.07	ON when an error occurs in a data exchange between the CPU Unit and a CPU Bus Unit (including an error in the CPU Bus Unit itself).
CPU Bus Unit Error, Unit Number Flags	A417.00 to A417.15	When an error occurs in a data exchange between the CPU Unit and a CPU Bus Unit, the CPU Bus Unit Error Flag (A402.07) and the corresponding flag in A417 are turned ON. Bits 00 to 15 correspond to unit numbers 0 to F.

10-4-3 Special I/O Unit Flags/Bits

● Restarting and Initializing Special I/O Units

With Special I/O Units, changes to initial settings for allocated DM Area words for the CPU Unit can be enabled by turning ON the following restart bits without turning OFF the PLC. Normally, this operation is performed with the user program.

Auxiliary Area Flags and Words

Name	Address	Description
Special I/O Unit Restart Bits	A502.00 to A507.15	Bits A502.00 to A507.15 correspond to Special I/O Units 0 to 95. Turn ON a bit to restart the corresponding Unit.
Special I/O Unit Initialization Flags	A330.00 to A335.15	These flags correspond to Special I/O Units 0 to 95. A flag will be ON while the corresponding Unit is initializing after the power is turned ON or the Unit's Restart Bit (A502.00 to A507.15) is turned ON. Bits A330.00 to A335.15 correspond to unit numbers 0 to 95.

● Special I/O Unit Errors

The following table gives error information related to Special I/O Units.

Auxiliary Area Flags and Words

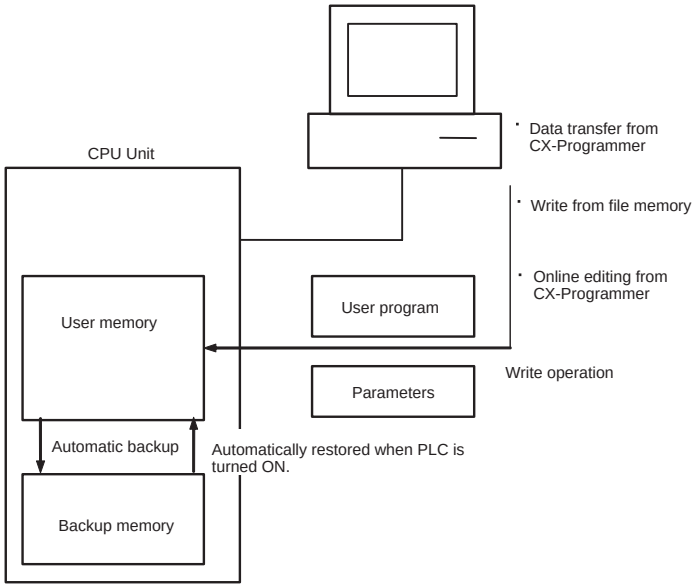
Name	Address	Description
Special I/O Unit Number Duplication Flags	A411.00 to A416.15	The Duplication Error Flag (A401.13) and the corresponding flag in A411 through A416 will be turned ON when a Special I/O Unit's unit number has been duplicated. Bits A411.00 to A416.15 correspond to unit numbers 0 to 95.
Special I/O Unit Setting Error Flag (non-fatal error)	A402.02	ON when an installed Special I/O Unit does not match the Special I/O Unit registered in the I/O table.
Special I/O Unit Setting Error, Unit Number Flags	A428.00 to A433.15	When a Special I/O Unit Setting Error occurs, A402.02 and the corresponding flag in these words are turned ON. Bits A428.00 to A433.15 correspond to unit numbers 0 to 95.
Special I/O Unit Error Flag (non-fatal error)	A402.06	ON when an error occurs in a data exchange between the CPU Unit and a Special I/O Unit (including an error in the Special I/O Unit itself).
Special I/O Unit Error, Unit Number Flags	A418.00 to A423.15	When an error occurs in a data exchange between the CPU Unit and a Special I/O Unit, the Special I/O Unit Error Flag (A402.06) and the corresponding flag in these words are turned ON.

10-5 Memory Management Functions

10-5-1 Automatic Backup

The user program and parameters are automatically backed up in backup memory whenever they are written to or altered in the CPU Unit.

- The following data is backed up automatically: User program, parameters (including the PLC name, PLC Setup, I/O tables, routing tables, and CPU Bus Unit data, such as the data link tables).
- The data is backed up automatically whenever the user program or parameters are written in the CPU Unit, including for data transfer operations from the CX-Programmer, online editing, data transfers from a Memory Card or EM file memory, etc.
- The user program and parameter data written to backup memory is automatically transferred to user memory in the CPU Unit at startup.



- The BKUP indicator on the front of the CPU Unit will light while data is being written to backup memory. When transferring the user program from the CX-Programmer or transferring data to the parameter area from file memory, do not turn OFF the power to the CPU Unit until the backup operation has been completed (i.e., until the BKUP indicator turns OFF). The following table shows the operations that will be performed if the power is turned OFF before the backup is completed.

● Operations after Power Is Interrupted during Automatic Backup

Automatic backup timing	Applicable data	Operation after power interruption
While a program is being transferred	User program	• The data will be lost. • A memory error will occur the next time the power is turned ON.
During online editing	User program: • Ladder tasks	• The changes made during online editing will be reflected the next time the power is turned ON. (If there is no Battery, however, the changes will not be reflected and a memory error will occur.)
	User program: • ST tasks • SFC tasks • FB definitions • SFC actions and transitions	• The contents of the online editing will be lost.
While the PLC Setup is being transferred, the I/O tables are being transferred, the I/O tables are being created or cleared, the routing tables is being transferred, or Special I/O Unit settings are being transferred.	Parameters	• The data will be lost. • A memory error will occur the next time the power is turned ON.

- If the power is turned OFF when there is a Battery installed in the CPU Unit and when only ladder tasks are being edited online, the status prior to the power interruption will be restored the next time the power is turned ON. Be careful, however, of the following points.
  - More time will be required to start the CPU Unit.
  - Even if there is a Battery in the CPU Unit, always be sure that the backup operation has been completed before turning OFF the power supply if the CPU Unit will be left unpowered for an extended period of time.
- The amount of time required to back up data (the time the BKUP indicator will be lit) will depend on the size of the user program, as shown in the following table.

User program size	Backup processing time	
	MONITOR mode	PROGRAM mode
100 Ksteps	20 s (cycle time of 3 ms)	3 s
250 Ksteps	47 s (cycle time of 7 ms)	6 s
400 Ksteps	75 s (cycle time of 11 ms)	10 s

Note The BKUP indicator will be lit when power is supplied to the CPU Unit.



Precautions for Correct Use

- Automatically back up the user program and parameter data to flash memory when they are written to the CPU Unit. I/O memory (including the DM, EM, Holding, and Auxiliary Areas), however, is not written to flash memory. The DM, EM, and Holding Areas can be held during power interruptions with a battery. If there is a battery error, the contents of these areas may not be accurate after a power interruption. If the contents of the DM, EM, Holding, and Auxiliary Areas are used to control external outputs, prevent inappropriate outputs from being made whenever the Battery Error Flag (A402.04) is ON.
- A backup memory error (non-fatal) will occur if an error in the CPU Unit backup memory is detected. If this error occurs, data will not be restored the next time the power supply is turned ON, and a memory error may occur. Therefore, it is recommended to save the data before turning OFF the power.

Additional Information

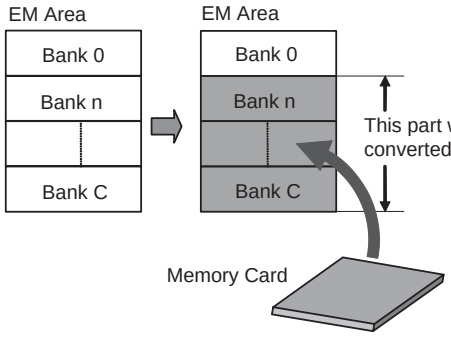
A backup status will be displayed in a Memory Backup Status Window by the CX-Programmer when backing up data from the CX-Programmer for transfer operations other than normal data transfers ***(PLC - Transfer)***. To obtain this window, setting to display the backup status dialog box must be checked in the PLC properties and the window must be selected from the View Menu. For normal transfer operations, the backup status will be displayed in the transfer window after the transfer status for the program and other data.

● Auxiliary Area Flags

Name	Address	Meaning
Backup Memory Error Flag	A315.15	Turns ON when the backup memory fails.

10-5-2 EM File Memory Functions

It is possible to use the EM Area instead of the Memory Card to save files in the CPU Unit. The banks after the specified starting bank are used as the file memory. For information on converting EM Area banks into file memory, refer to *7-1-2 Initializing File Memory*.



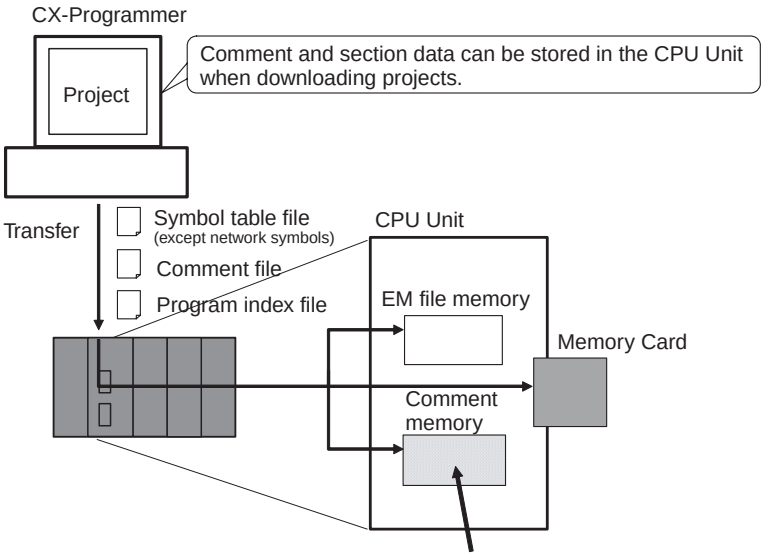
1. Specify the starting bank in the EM Area settings from the CX-Programmer.
2. When you transfer the program from the CX-Programmer, you can use the EM file memory.

You can select to convert part of the EM Area to either file memory or trace memory. If you convert it to file memory, you can use the converted part of the EM Area just like a Memory Card.

10-5-3 Comment Memory

A comment memory is provided within the CPU Unit's internal backup memory. The following comment/section data can be stored in and read from comment memory even if neither Memory Card nor EM file memory are available.

- Symbol table files (including CX-Programmer symbols and I/O comments, but not network symbols)
- Comment files (CX-Programmer rung comments and other comments)
- Program index files (CX-Programmer section names, section comments, and program comments)

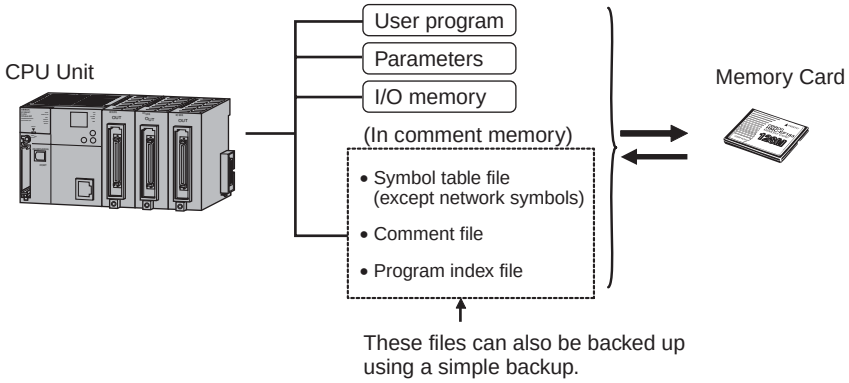


When downloading projects, either of the following storage locations can be selected as the transfer destination for comment data and section data.

- Comment memory (in CPU Unit's backup flash memory)
- Memory Card
- EM file memory

The following files stored in comment memory can be backed up to a Memory Card when a simple backup operation is executed, or the files can be restored to comment memory from the Memory Card.

- Symbol table files (including CX-Programmer symbols and I/O comments, but not network symbols)
- Comment files (CX-Programmer rung comments and other comments)
- Program index files (CX-Programmer section names, section comments, and program comments)

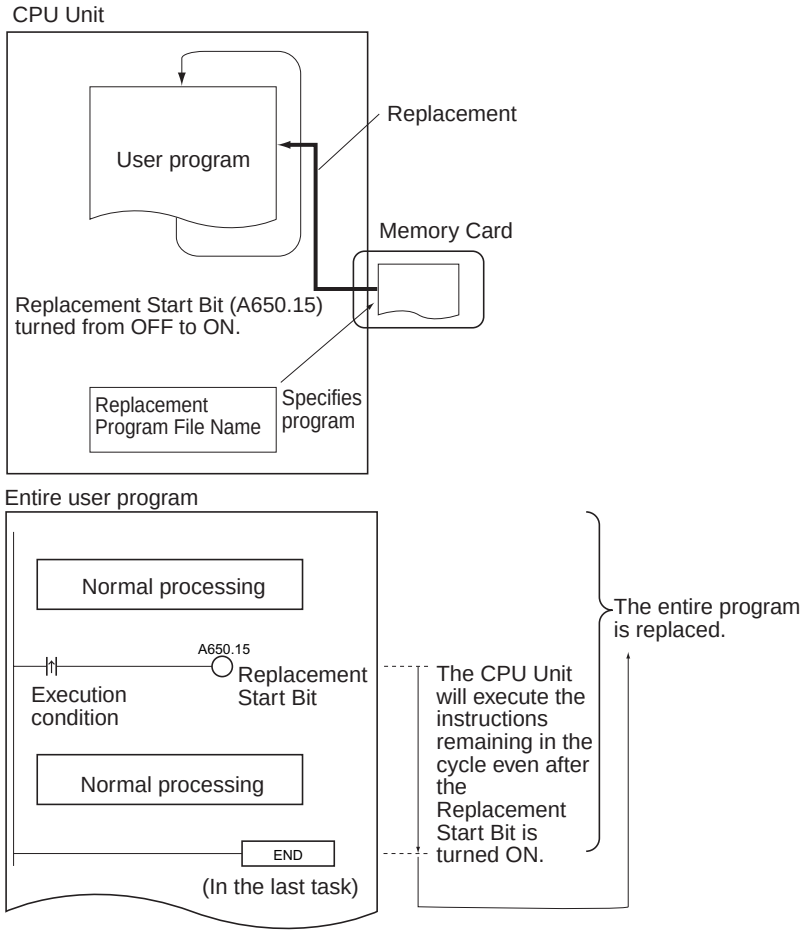


This enables backup/restoration of all data in the CPU Unit including I/O comments if an error occurs or when adding a CPU Unit with the same specifications without requiring the CX-Programmer.

10-5-4 Replacing the Entire Program during Operation

Overview

The entire task program can be replaced during operation in RUN or MONITOR mode by turning ON the Replacement Start Bit (A650.15). The program/network symbol file (.OBJ) with a file name specified in advance in the Auxiliary Area will be read from the Memory Card and it will replace the program at the end of the current cycle.



The program can also be replaced when program execution is stopped (PROGRAM mode) by turning ON the Replacement Start Bit from the CX-Programmer.

- The replacement program file cannot be read from EM file memory.
- The Replacement Start Bit (A650.15) can be turned ON at any location (program address) in the program.
- The CPU Unit will execute the instructions remaining in the cycle after the Replacement Start Bit goes from OFF to ON.
- The program will not be executed while the program is being replaced. After the program has been replaced, operation will be started again just as if the CPU Unit were switched from PROGRAM mode to RUN or MONITOR mode.
- The program will be replaced at the end of the cycle in which the Replacement Start Bit was turned from OFF to ON, i.e., after END(001) is executed in the last task in the program.

CPU Operation during Program Replacement

The CPU Unit's operation will be as follows during program replacement:

- Program execution: Stopped
- Cycle time monitoring: No monitoring

● Operations Continuing during and after Program Replacement

- When the IOM Hold Bit (A500.12) is ON, the data in the following memory areas will be maintained: the CIO Area, Work Area (W), Timer Completion Flags (T), Index Registers (IR), Data Registers (DR), and the current EM bank number.

Note Timer PVs will be cleared during program replacement.

- If the IOM Hold Bit is ON when the program is transferred, loads that were being output before program replacement will continue to be output after replacement. Be sure that external loads will operate properly after program replacement.
- The status of force-set and force-reset bits will be maintained through the program replacement if the Forced Status Hold Bit (A500.13) is ON.
- Interrupts will be masked.
- If data tracing is being performed, it will be stopped.
- Differentiation Flags will be initialized whether the IOM Hold Bit is ON or OFF.

● Operations after Program Replacement

- The status of the cyclic tasks depends upon their operation-start properties. (Their status is the same as it would be if the PLC were switched from PROGRAM to RUN/MONITOR mode.)
- The First Cycle Flag (A200.11) will be ON for one cycle after program execution resumes. (The status is the same as it would be if the PLC were switched from PROGRAM to RUN/MONITOR mode.)

Procedure

1 Mount a Memory Card containing the program file in the CPU Unit.

2 Set the Program File Name (A654 to A657) and Program Password (A651) in the Auxiliary Area, and then turn ON the Replacement Start Bit (A650.15).



Additional Information

Turn ON the IOM Hold Bit (A500.12) if you want to maintain the status of I/O memory data through the program replacement.

Turn ON the Forced Status Hold Bit (A500.13) if you want to maintain the status of force-set and force-reset bits through the program replacement.



Precautions for Safe Use

If the IOM Hold Bit (A500.12) is ON before the program is replaced, the status of bits in I/O memory will be maintained after program replacement. Be sure that external loads will operate properly with the same I/O memory data.

If the Forced Status Hold Bit (A500.13) is ON before the program is replaced, the forced status in I/O memory will be maintained after program replacement. Be sure that external loads will operate properly with the same forced status.

Replacement File

The specified program/network symbol file will be read from the Memory Card.

File	File name and extension	Turning ON the Replacement Start Bit (A650.15) in the Auxiliary Area	Specifying the replacement file name (********)
Program/network symbols file	********.OBJ	Replacing the CPU Unit's user program	Write the replacement program file name to A654 through A657 before program replacement.

Conditions Required for Program Replacement

The following conditions are required in order to replace the program during operation.

- The program/network symbol file specified in the Program File Name words (A654 to A657) exists in the Memory Card's root directory.
- The Memory Card has been detected by the CPU Unit. (A343.15 must be ON.)
- No fatal errors have occurred.
- No file memory operations are being executed. (A343.13 must be OFF.)
- Data is not being written to the Program Area.
- The access right is available. (For example, data is not being transferred from the CX-Programmer to the PLC.)

Note The program may be transferred in any operating mode.

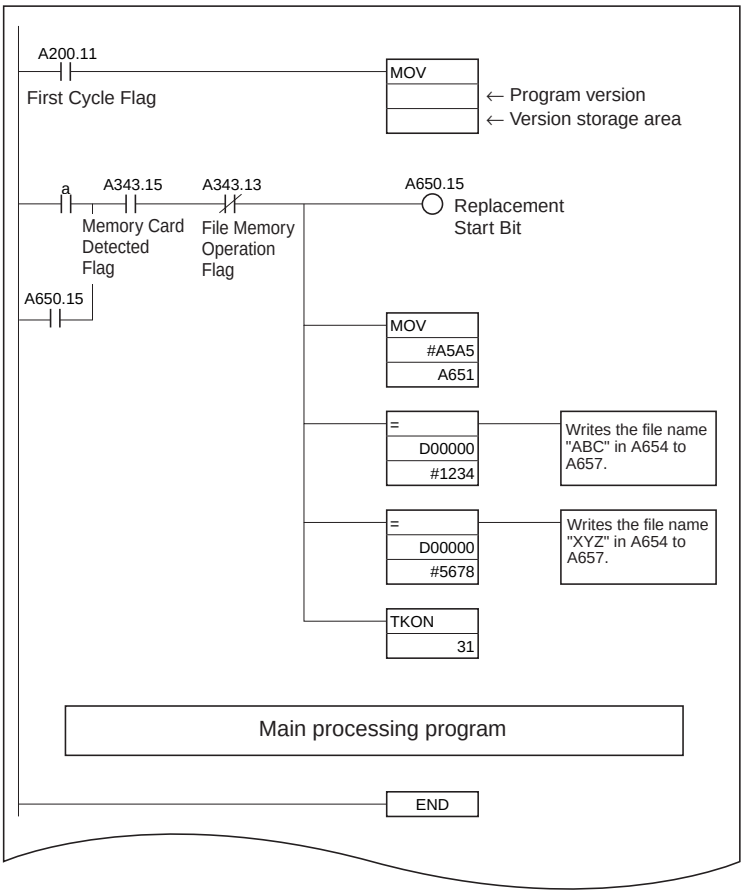
Example Programs

Example Program 1

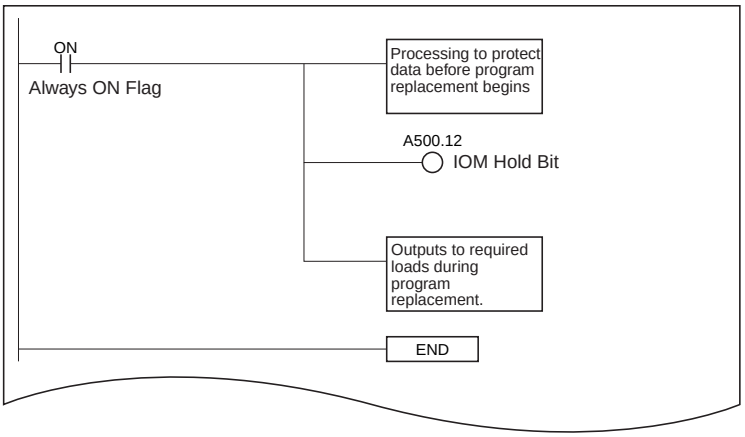
In the following example, program/network symbol files ABC.OBJ and XYZ.OBJ are stored in the Memory Card and one program or the other is selected depending upon the value of D0. D0 is set to #1234 to select ABC.OBJ or to #5678 to select XYZ.OBJ.

Another task is started to perform any processing required before program replacement, including set the IOM Hold Bit.

Main Task (Cyclic Task 0)

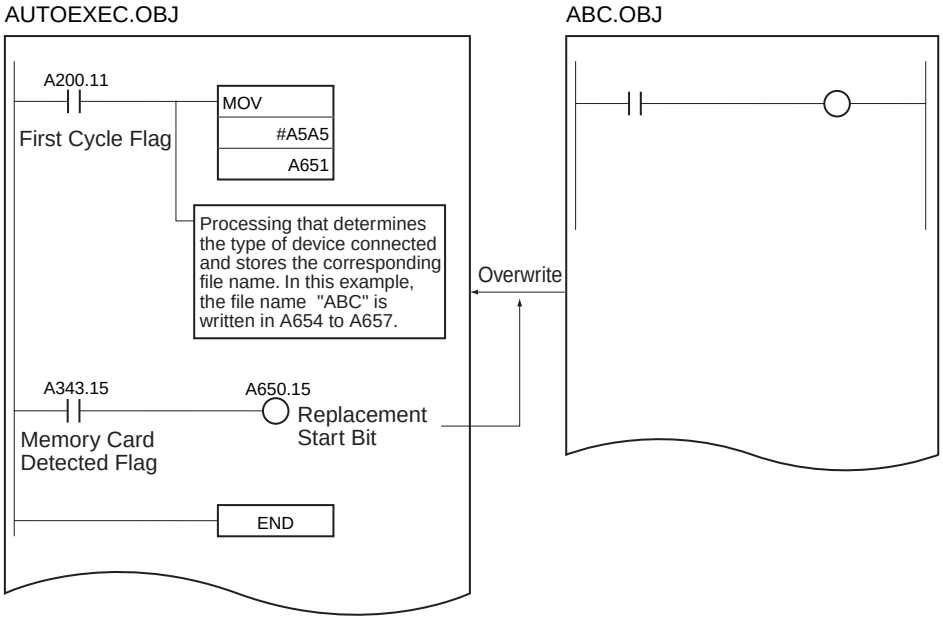


Task Protecting Data during Program Replacement
(Cyclic task 31, standby status at startup)



Example Program 2

In this example, the program/network symbol files for several devices and the program/network symbol file for automatic transfer at startup (AUTOEXEC.OBJ or REPLACE.OBJ) are stored in a Memory Card. When the PLC is turned ON, the automatic transfer at startup file is read and then that program is replaced later with a program files for different devices.



Auxiliary Area Flags and Words

Name	Address	Operation
File Memory Operation Flag	A343.13	ON while any of the following operations is being executed. OFF when none of them are being executed. • Memory Card detection • The CPU Unit has sent a FINS command to itself using CMND(490). • FREAD(700) or FWRIT(701) is being executed. • The program is being overwritten using an Auxiliary Area control bit (A650.15). • A simple backup operation is being performed.
Memory Card Detected Flag	A343.15	ON when a Memory Card has been detected. OFF when a Memory Card has not been detected.
IOM Hold Bit	A500.12	Turn ON this bit to preserve the status of the I/O Memory when changing the CPU Unit between PROGRAM and RUN or MONITOR mode or turning ON the power supply. ON: I/O memory status retained when changing the operating mode. OFF: I/O memory status cleared when changing the operating mode.
Forced Status Hold Bit	A500.13	Turn ON this bit to preserve the status of bits that have been force-set or force-reset when changing the CPU Unit between PROGRAM and RUN or MONITOR mode or turning ON the power supply.
Replacement Completion Code	A650.00 to A650.07	• Codes for normal program replacement (A650.14 OFF): 01 Hex: The program/network symbols file (.OBJ) replaced the program. • Codes for incomplete program replacement (A650.14 ON): 00 hex: A fatal error occurred. 01 hex: A memory error occurred. 11 hex: The program is write-protected. 12 hex: The program password in A651 is incorrect. 21 hex: A Memory Card is not installed. 22 hex: The specified file does not exist. 23 hex: The specified file is too large (memory error). 31 hex: One of the following operations was being performed: - A file memory operation was being performed. - The program was being written. - The operating mode was being changed.
Replacement Error Flag	A650.14	Turned ON when an error occurred while trying to replace the program after A650.15 was turned from OFF to ON. Turned OFF the next time that A650.15 is turned from OFF to ON again.

Name	Address	Operation
Replacement Start Bit	A650.15	If this bit has been enabled by setting the Program Password (A651) to A5A5 hex, program replacement will start when this bit is turned from OFF to ON. Do not turn this bit from OFF to ON again during program replacement. This bit is automatically turned OFF when program replacement is completed (normally or with an error) or the power is turned ON. The status of this bit can be read from the CX-Programmer, PT, or host computer to determine whether program replacement has been completed or not.
Program Password	A651	Write the password to this word to enable program replacement. A5A5 hex: Enables the Replacement Start Bit (A650.15). Other value: Disables the Replacement Start Bit (A650.15). This bit is automatically turned OFF when program replacement is completed (normally or with an error) or the power is turned ON.
Program File Name	A654 to A657	Before starting program replacement, write the file name of the replacement program file in these words in ASCII. Just write the 8-character filename; the OBJ extension is added automatically. Write the characters in order from A654 (most significant byte first). If the file name has fewer than 8 characters, pad the remaining bytes with space codes (20 hex). Do not include any NULL characters or spaces within the file name itself. The following example shows the data for the program file ABC.OBJ: 15 0 A654 41 42 A655 43 20 A656 20 20 A657 20 20

10-6 Security Functions

The following protection can be applied to the CJ-series user programs.

10-6-1 Write-protection Using the DIP Switch

Write-protection can be set so that Data in the user program and parameter area (e.g., PLC Setup and I/O tables) cannot be overwritten from the CX-Programmer. This function can prevent the program from being overwritten inadvertently.

To set write protection, turn ON pin 1 on the CPU Unit's DIP switch. Even when this function is enabled, it is still possible to read and display the program using the CX-Programmer.

● DIP Switch on Front of CPU Unit

Pin No.	Name	Setting
1	User program memory write protection	ON: Protection ON OFF: Protection OFF

10-6-2 Read Protection Using Passwords

Read protection using passwords can be set. User programs, specific tasks, and function blocks for protected programs cannot be displayed or edited unless the user enters the password in the CX-Programmer.

Types of Protection

The following types of read protection using passwords can be used.

● UM Read Protection

Reading the user program from the CPU Unit will be disabled.

● Task Read Protection

Displaying tasks in the CX-Programmer will be disabled for any one or more tasks. It will be possible to read the user program from the CPU Unit, but the protected tasks will not be displayed on the CX-Programmer.

Additional Information

- The entire program can be transferred to another CPU Unit even if individual tasks in the program are read-protected. The task read protection remains effective for the password-protected tasks.
- When the CX-Programmer is used to compare a user program in the computer's memory with a user program in the CPU Unit, password-protected tasks will be compared too.

● Function Block Protection

The following protection can be applied to the desired function blocks.

- Write/Display Protection
  Displaying function blocks will be disabled in the CX-Programmer, so it will not be possible to make changes.

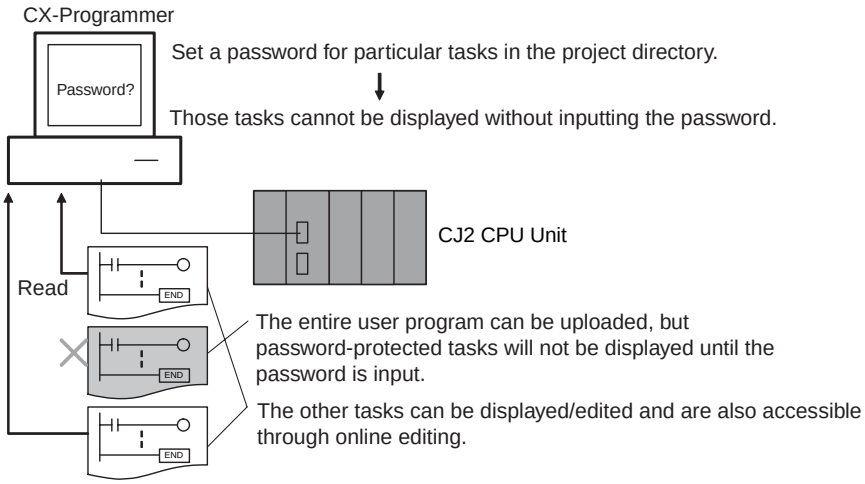
- Write Protection
  Displaying function blocks will be enabled in the CX-Programmer, but it will not be possible to make changes.

Additional Information

Even if task read protection is applied, it will be possible to read the function block definitions if a user program that includes function blocks is used. To read-protect the function block, use function block protection.

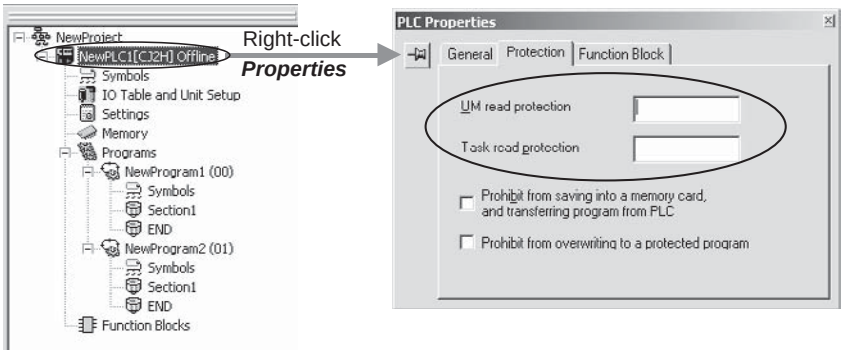
Additional Information

Task read protection cannot be set if UM read protection is already set. However, it is possible to set UM read protection after task read protection has been set.



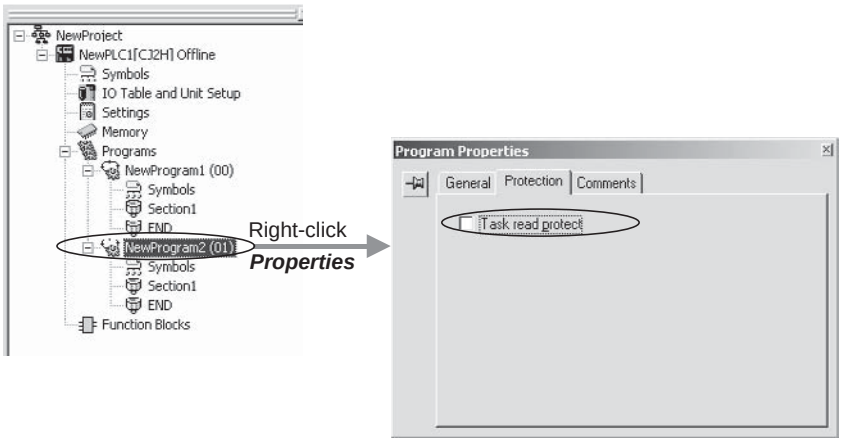
Operating Procedure

1 Display the Protection Tab Page of the PLC Properties Windows and register a password for protection.

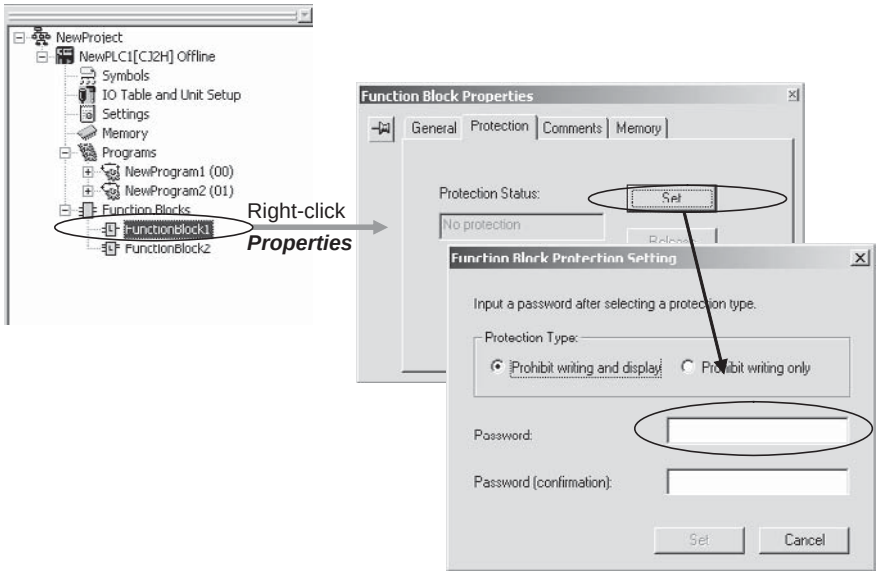


- *UM read protection* Field
  Enter a password for UM read protection.
- *Task read protection* Field
  Enter a password for task read protection.

2 To set read protection for a task, select the task and then select the *Task read protect* Check Box on the *Program Properties* Dialog Box.



3 To apply read protection to function block (FB) definitions, select the function block to be protected, click the **Set** Button in the function block properties, and enter a password in the *Function Block Protections Setting* Field.



4 Connect online and execute either step a or b below.

(a) Transferring the Program and Setting Password Protection:
Select ***PLC - Transfer - To PLC*** to transfer the program. The tasks registered in step 2 will be password-protected.

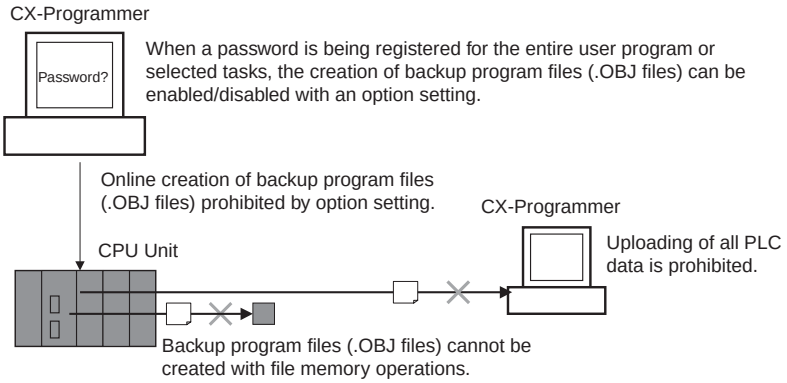
(b) Setting Password Protection without Transferring the Program:
Select ***PLC - Protection - Set Password*** and click the **OK** button. The tasks registered in step 2 will be password-protected.

Enabling/Disabling Saving to Memory Cards

Overview

When the entire program or individual tasks are read-protected from the CX-Programmer, an option can be set to enable or disable creating or backing up .OBJ program/network symbol files. It will not be possible to create program/network symbol files (.OBJ files) with file memory operations if creating and backing up program/network symbol files is prohibited with this setting. (This setting prohibits both online transfers to a Memory Card/EM file memory as well as offline storage of PLC data that was uploaded to the CX-Programmer.)

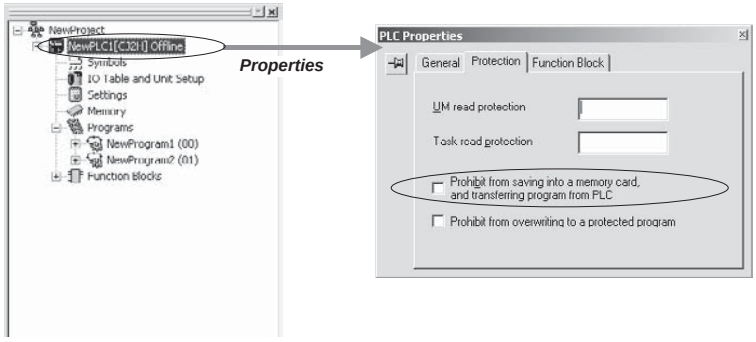
Disabling the creation of file memory program files can help prevent illegal copying of user programs.



- The simple backup operation can still be performed when creating program/network symbol files is prohibited, but the backup program/network symbol file (BACKUP.OBJ) will not be created.
- The program can be copied when program read protection is not enabled.
- The setting to enable/disable creating program/network symbol files will not take effect unless the program is transferred to the CPU Unit. Always transfer the program after changing this setting.

Operating Procedure

1 To register a password in the *UM read protection* Box or *Task read protection* Box, select the *Prohibit from saving into a memory card, and transferring program from PLC* Check Box in the **Protection** Tab Page in the PLC Properties Dialog Box of the CX-Programmer.



2 Either select ***PLC - Transfer - To PLC*** to transfer the program or select ***PLC - Protection - Set Password*** and click the **OK** Button.

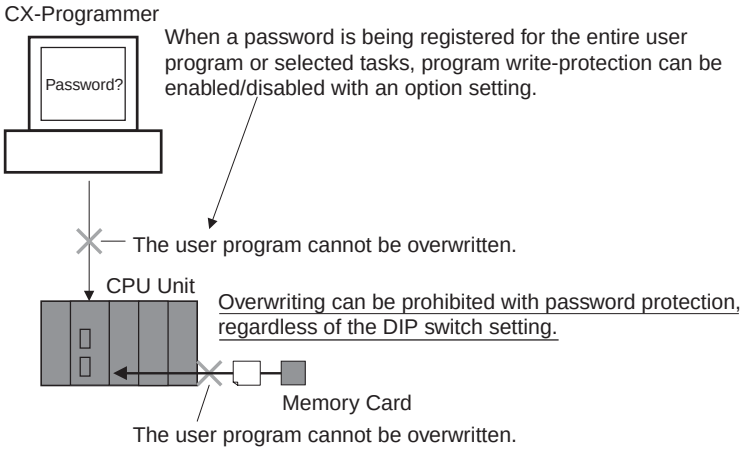
Usage

- This option can be used to prevent the program from being transferred out of the PLC using the password.

Enabling and Disabling Program Overwriting

● Overview

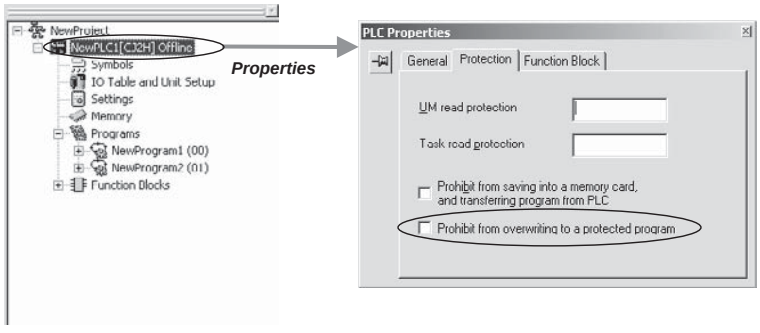
In addition to using the DIP switch as described above, the program (or selected tasks) can also be write-protected if the write protection option is selected from the CX-Programmer when a password is being registered for the entire program or those selected tasks. The write protection setting can prevent unauthorized or accidental overwriting of the program.



- If the selected tasks or program are write-protected by selecting this option when registering a password, only the tasks (program) that are password-protected will be protected from overwriting. It will still be possible to overwrite other tasks or programs with operations such as online editing and task downloading.
- All tasks (programs) can be overwritten when program read protection is not enabled.
- The setting to enable/disable creating program files will not take effect unless the program is transferred to the CPU Unit. Always transfer the program after changing this setting.

● Operating Procedure

1 To register a password in the *UM read protection* Box or *Task read protection* Box, select the *Prohibit from overwriting to a protected program* Option in the **Protection** Tab Page in the PLC Properties Dialog Box of the CX-Programmer.



2 Either select ***PLC - Transfer - To PLC*** to transfer the program or select ***PLC - Protection - Set Password*** and click the **OK** button.

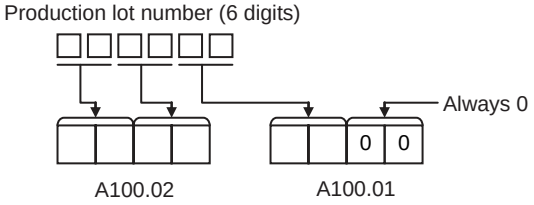
Auxiliary Area Flags and Bits related to Password Protection

Name	Bit address	Description
UM Read Protection Flag	A99.00	Indicates whether or not the PLC (the entire user program) is read-protected. OFF: UM read protection is not set. ON: UM read protection is set.
Task Read Protection Flag	A99.01	Indicates whether or not selected program tasks are read-protected. OFF: Task read protection is not set. ON: Task read protection is set.
Program Write Protection for Read Protection	A99.02	Indicates whether or not the write protection option has been selected to prevent overwriting of programs. OFF: Overwriting allowed ON: Overwriting prohibited (write-protected)
Enable/Disable Bit for Program Backup	A99.03	Indicates whether or not a backup program/network symbol file (.OBJ file) can be created when UM read protection or task read protection is set. OFF: Creation of backup program file allowed ON: Creation of backup program file prohibited

10-6-3 Program Operation Protection Using Production Lot Numbers

The program can be protected against operation by using the production lot number stored in words A100.01 and A100.02 of the Auxiliary Area. The production lot number cannot be changed by the user.

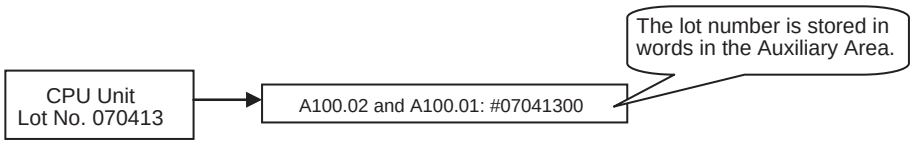
- The production lot number is six digits. The leftmost four digits are stored in A100.02 and the rightmost two digits are stored in A100.01.

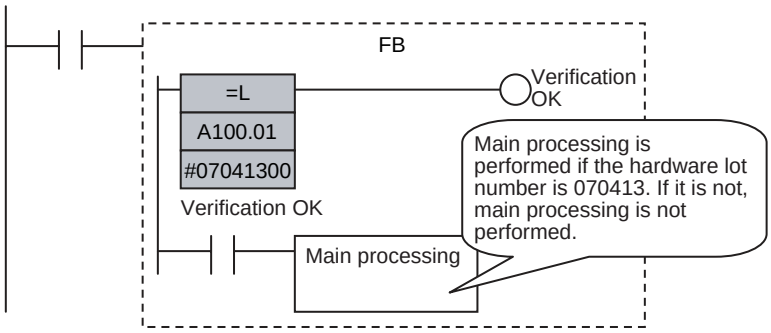


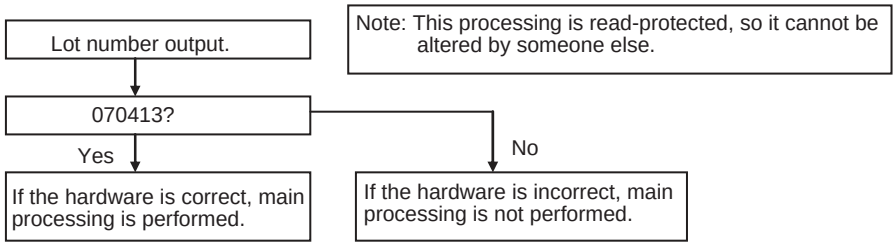
Example of Production Lot Number Storage

Production date	Production lot number	A100.02	A100.01
June 20, 2008	080620	0806	2000

Application Example: Operating the Program Only for a CPU Unit with a Specific Production Lot Number







10-6-4 Write Protection from FINS Commands

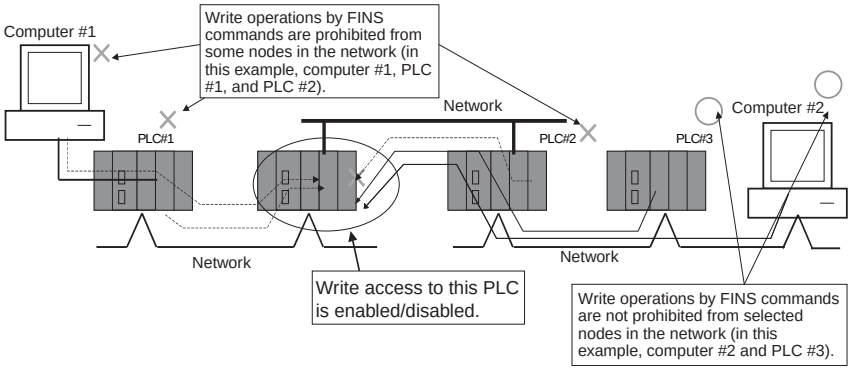
It is possible to prohibit write operations and other editing operations sent to the PLC's CPU Unit as FINS commands through a network (including write operations from CX-Programmer, CX-Protocol, CX-Process, and other applications using FinsGateway, but excluding a direct serial connection). Read processes are not prohibited.

FINS write protection can disable write processes such as downloading the user program, PLC Setup, or I/O memory, changing the operating mode, and performing online editing.

It is possible to exclude selected nodes from write protection so that data can be written from those nodes.

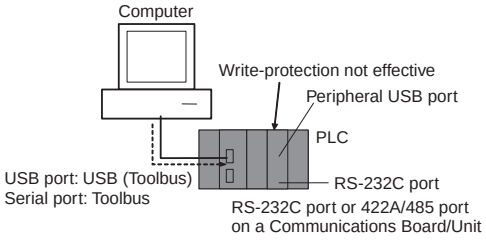
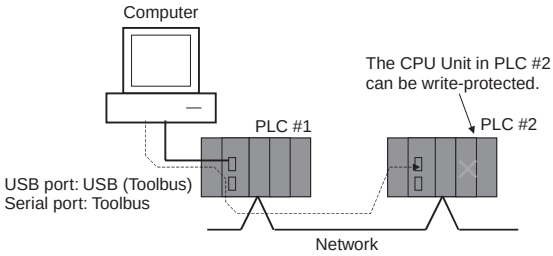
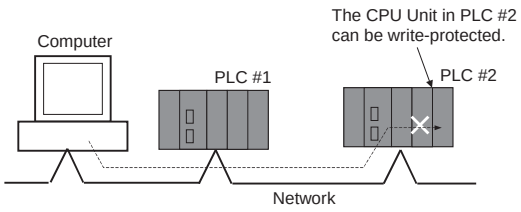
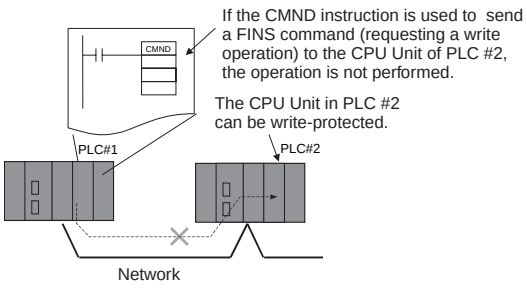
An event log in the CPU Unit automatically records all write processes sent through the network and that log can be read with a FINS command.

Example:



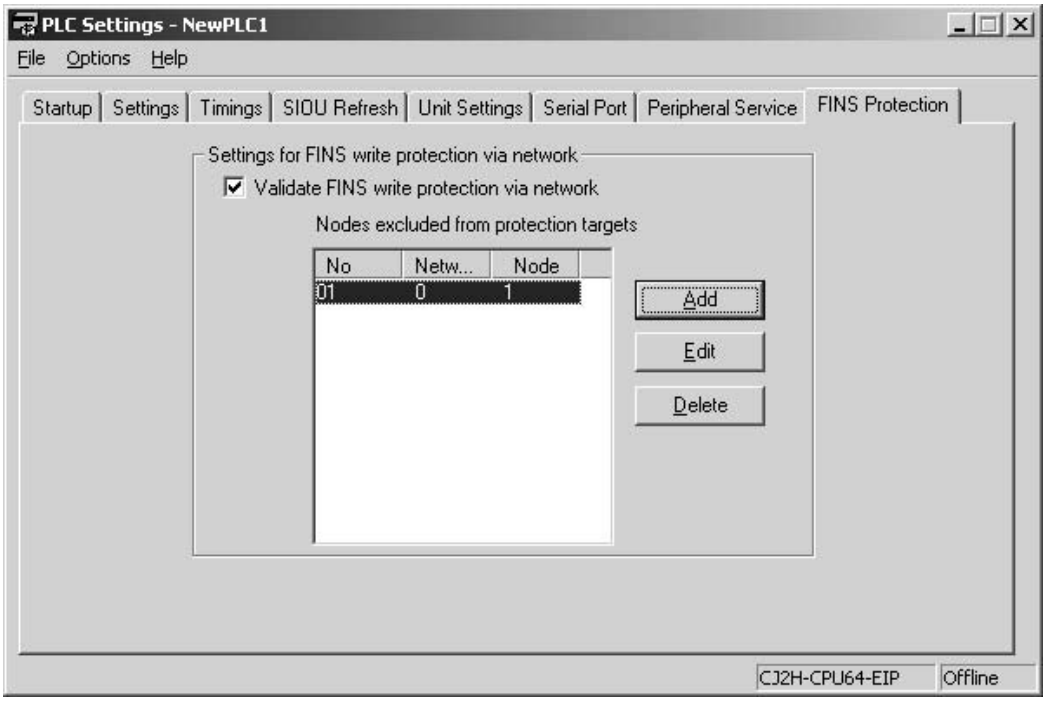
Note This function prohibits writing by FINS commands only, so it has no effect on write operations by functions other than FINS commands, such as data links.

Example Write Protection Patterns

Connection pattern		Diagram (example)	Write protection
From a computer through a direct serial connection	Direct connection to PLC		Cannot be applied.
	Gateway connection (Serial-to-network) to PLC		Can be applied.
From a computer through a direct network connection			Can be applied.
From another PLC in the network			Can be applied.

Operation

With the CX-Programmer, open the PLC Setup's *FINS Protection* Tab and select the *Validate FINS write protection via network* Option. When this option is selected, it will not be possible to execute write operations for that CPU Unit with FINS commands sent through a network (except a direct serial connection). To permit write operations from particular nodes, enter network addresses and node addresses for the node under *Nodes excluded from protection targets*. (Up to 32 nodes can be excluded from FINS Write Protection).

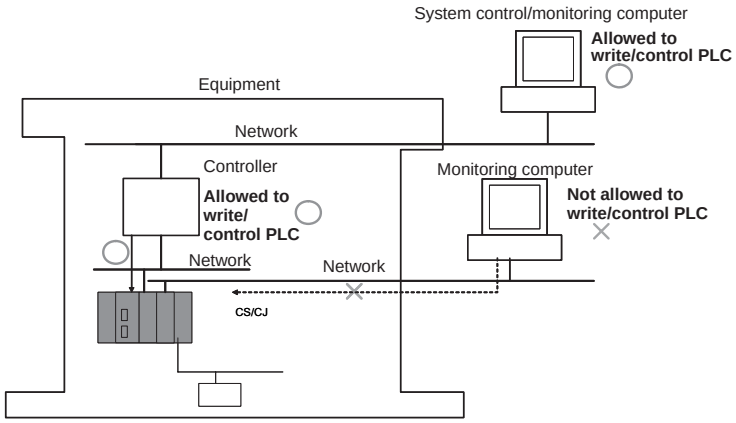


PLC Setup

Item	Description	Settings	Default setting
Validate FINS write protection via network	Sets whether the CPU Unit is write-protected from FINS commands sent through the network. (Does not prohibit FINS commands sent through a direct serial connection).	OFF: Write protection disabled ON: Write protection enabled	OFF: Write protection disabled
Nodes Excluded from Write Protection	This area lists the nodes in the network that are not restricted by FINS write protection. Up to 32 nodes can be specified. These settings are effective only when FINS write protection is enabled.		
	Network address: Network address of the FINS command source	00 to 7F hex	
	Node address: Node address of the FINS command source	01 to FE hex, or FF hex (FF hex: node address unspecified)	
Number of Nodes Excluded from FINS Write Protection (Do not set this value. It is automatically calculated by the CX-Programmer.)	Contains the number of nodes that are not subject to the FINS write protection. If 0 is specified (no nodes excluded from write protection), FINS write commands are prohibited from all nodes other than the local node. This setting is effective only when FINS write protection is enabled.	0 to 32 (00 to 20 hex) (A value of 0 indicates that all nodes are subject to write protection.)	0 (All nodes subject to write protection.)

Usage

- The system can be configured so that a PLC can be written only from authorized nodes in the network. (For example, use this function when the system's control/monitoring computer is the only node allowed to write to a Controller within a piece of equipment.)
  By limiting the number of nodes that can write to the PLC, it is possible to prevent system problems caused by unintentional overwrites during data monitoring.



Operations Restricted by Network FINS Write Protection

FINS Write Commands

The following FINS commands are restricted by FINS write protection when sent to the CPU Unit through the network.

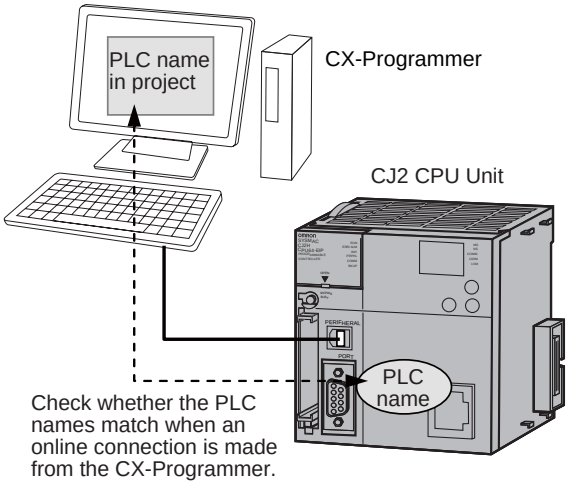
Code	Command name
0102 hex	MEMORY AREA WRITE
0103 hex	MEMORY AREA FILL
0105 hex	MEMORY AREA TRANSFER
0202 hex	PARAMETER AREA WRITE
0203 hex	PARAMETER AREA FILL (CLEAR)
0307 hex	PROGRAM AREA WRITE
0308 hex	PROGRAM AREA CLEAR
0401 hex	RUN
0402 hex	STOP
0702 hex	CLOCK WRITE
0C01 hex	ACCESS RIGHT ACQUIRE

Code	Command name
2101 hex	ERROR CLEAR
2103 hex	ERROR LOG POINTER CLEAR
2203 hex	SINGLE FILE WRITE
2204 hex	FILE MEMORY FORMAT
2205 hex	FILE DELETE
2207 hex	FILE COPY
2208 hex	FILE NAME CHANGE
220A hex	MEMORY AREA-FILE TRANSFER
220B hex	PARAMETER AREA-FILE TRANSFER
220C hex	PROGRAM AREA-FILE TRANSFER
2215 hex	CREATE/DELETE DIRECTORY
2301 hex	FORCED SET/RESET
2302 hex	FORCED SET/RESET CANCEL

10-6-5 PLC Names

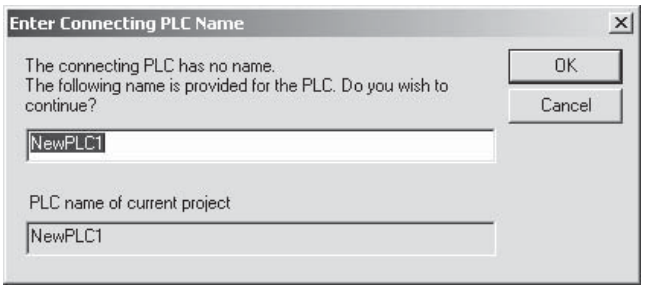
PLC Names

You can check whether the PLC name in the project matches the PLC name of the CPU Unit at the connection target when an online connection is made from the CX-Programmer. This enables preventing incorrect connection from the CX-Programmer.



Registering PLC Names

The Enter Connecting PLC Name Dialog Box will be displayed when online connection is made to a CJ2 CPU Unit that is using default settings or that has had memory all cleared.

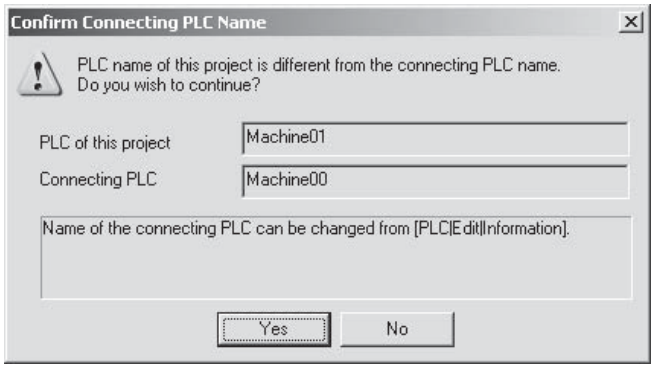


Enter the PLC name to register to the CJ2 CPU Unit connected online, and then click the **OK** Button. If a PLC name will not be entered, click the **Cancel** Button.

PLC Name Mismatch

If PLC names are saved in the CJ2 CPU Units, the system will automatically check whether the PLC name for the CPU Unit at the connection target matches the name of the PLC in the project when an online connection is made.

The following warning will be displayed if the PLC name saved in the CJ2 CPU Unit at the connection target is different from the PLC name in the project. Select whether to continue with the online connection.



- Yes Button
  Click the **Yes** Button to continue with the online connection with different names.
- No Button
  Click the **No** Button to go offline.

Additional Information

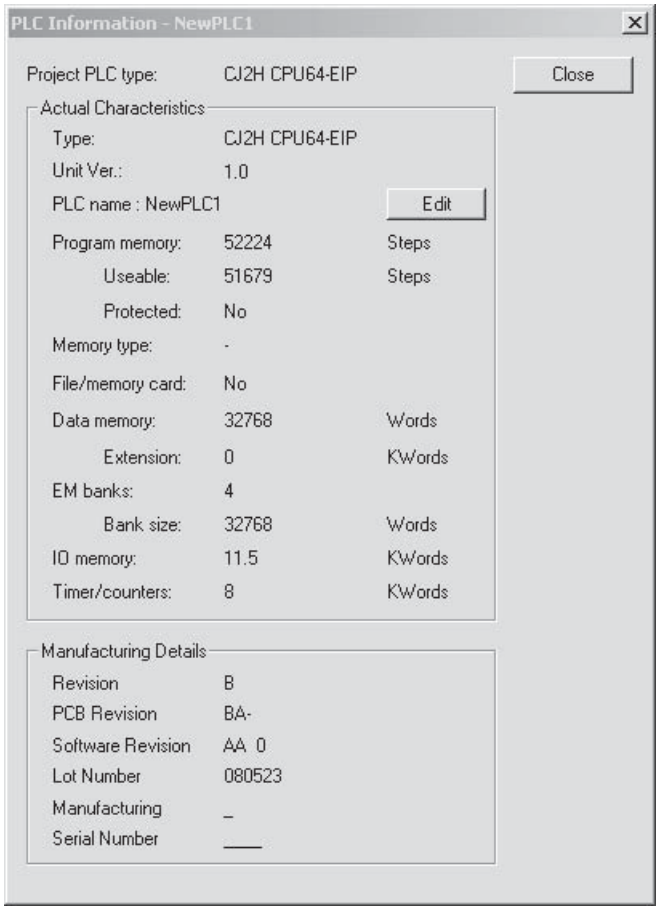
To not automatically check whether PLC names match, clear the selection of the *Check the connecting PLC name when going online* Check Box in the PLC Properties Dialog Box.

Changing PLC Names

Use the following procedure to change the PLC name saved in a CJ2 CPU Unit.

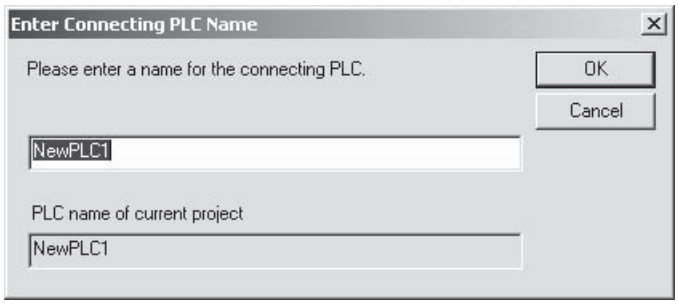
1 Connect online with the CX-Programmer, and select ***Edit*** – ***Information*** from the PLC Menu.

The following dialog box will be displayed.



2 Click the Edit Button to the right of the PLC Name Area.

The following Enter Connecting PLC Name Dialog Box will be displayed.



3 Enter the PLC name to register to the connection target PLC, and then click the **OK** Button.

10-7 Debugging

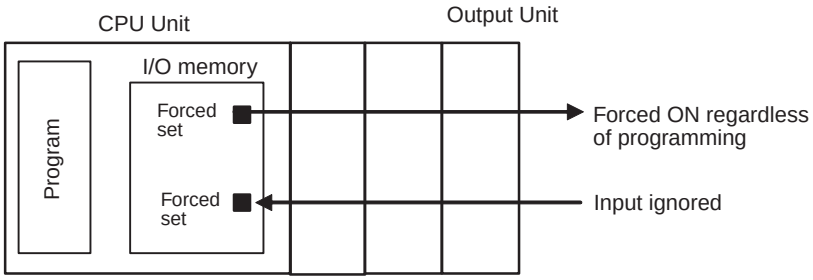
10-7-1 Forced Set/Reset

The CX-Programmer can force-set (ON) or reset (OFF) specified bits (CIO Area, Auxiliary Area, HR Area, and timer/counter Completion Flags). Forced status will take priority over status output from the program or I/O refreshing. This status cannot be overwritten by instructions, and will be stored regardless of the status of the program or external inputs until it is cleared from the CX-Programmer.

Force-set/reset operations are used to force input and output during a trial operation or to force certain conditions during debugging.

Force-set/reset operations can be executed in either MONITOR or PROGRAM modes, but not in RUN mode.

Note Turn ON the Forced Status Hold Bit (A500.13) and the IOM Hold Bit (A500.12) at the same time to retain the status of bits that have been force-set or reset when switching the operating mode.
Turn ON the Forced Status Hold Bit (A500.13) and the IOM Hold Bit (A500.12), and set the Forced Status Hold Bit at Startup setting in the PLC Setup to retain the status of the Forced Status Hold Bit at startup. This will enable holding forced status when turning ON the power.



Bits in the following areas can be force-set and reset.

CIO Area (I/O Area bits, Data Link Area bits, CPU Bus Unit Area bits, Special I/O Unit Area bits, Internal I/O Area bits), Work Area, Timer Completion Flags, Holding Area, Counter Completion Flags, and the banks in the EM Area for which using the EM Area force-setting/resetting function is specified*1, *2

*1 The EM Area force-setting/resetting function is supported by CJ2H CPU Units with unit version 1.2 or later and CJ2M CPU Units. CX-Programmer version 8.3 is also required.

*2 For CJ2H CPU Units, force-setting/resetting bits in the EM Area is also possible for any of the following banks for which automatic address allocation is specified.
CJ2H-CPU64/65(-EIP): EM bank 3 hex
CJ2H-CPU66(-EIP): EM banks 6 to 9 hex
CJ2H-CPU67(-EIP): EM banks 7 to E hex
CJ2H-CPU68(-EIP): EM banks 11 to 18 hex

● CX-Programmer Operation

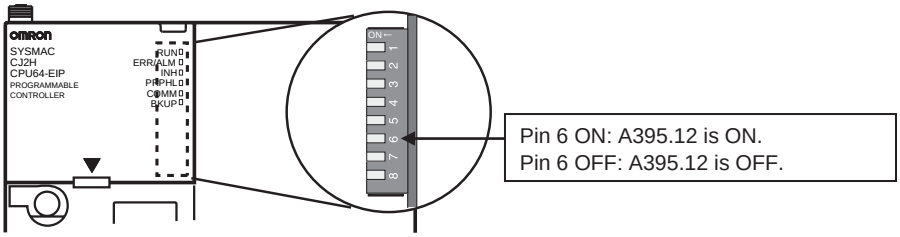
- Select bits for forced setting/resetting.
- Select forced set or forced reset.
- Clear forced status (including clearing all forced status at the same time).

● Forced Status Hold Bit

Name	Address	Function	Access
Forced Status Hold Bit	A500.13	Turn this bit ON to preserve the status of bits that have been force-set or force-reset when changing between PROGRAM and RUN or MONITOR mode or when turning ON the power.	Read/write

10-7-2 Test Input

The ON/OFF status of the DIP switch pin 6 on the front of the CPU Unit is stored in the DIP Switch Pin Status Flag (A395.12) in the Auxiliary Area. For debugging or other purposes, an input can be manipulated manually as a test without using an Input Unit.

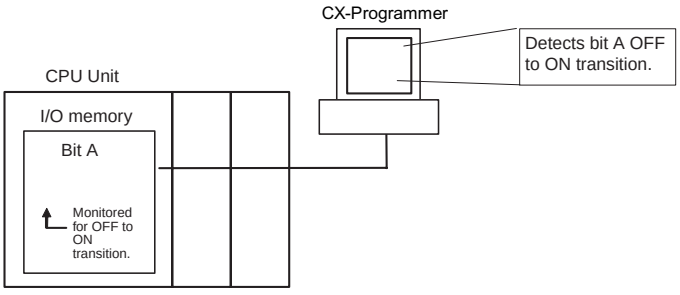


● Auxiliary Area Flags and Words

Name	Address	Function	Access
DIP Switch Pin Status Flag	A395.12	Contains the status set on pin 6 of the CPU Unit's DIP switch. (Refreshed every cycle.)	Read only

10-7-3 Differential Monitoring

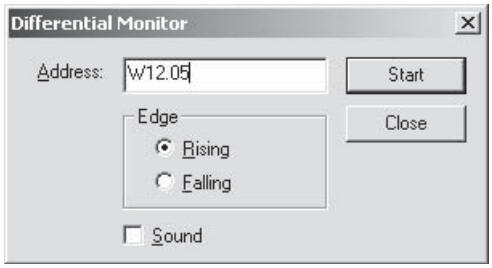
When the CPU Unit detects that a bit set by the CX-Programmer has changed from OFF to ON or from ON to OFF, the results are indicated in the Differentiate Monitor Completed Flag (A508.09). The Flag will turn ON when conditions set for the differential monitor have been met. The CX-Programmer can monitor and display these results on screen.



● Operation from CX-Programmer

1 Right-click the bit for differential monitoring.

2 Click ***Differential Monitor*** from the PLC Menu. The Differential Monitor Dialog Box will be displayed.



3 Click ***Rising*** or ***Falling***.

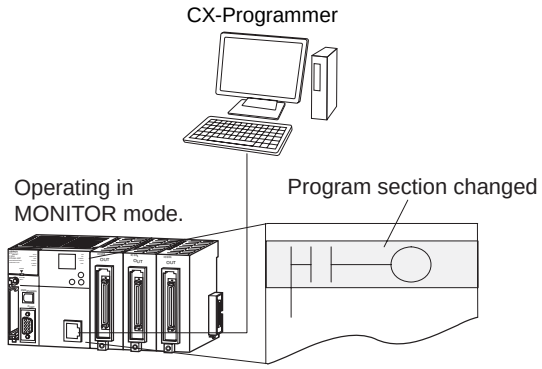
4 Click the **Start** button. The buzzer will sound when the specified change is detected and the count will be incremented.

5 Click the **Stop** button. Differential monitoring will stop.

10-7-4 Online Editing

The Online Editing function is used to add to or change part of a program in a CPU Unit directly from the CX-Programmer when the CPU Unit is in MONITOR or PROGRAM mode. One or more program sections are added or changed at a time from the CX-Programmer. The function is thus designed for minor program changes without stopping the CPU Unit.

Online editing is possible simultaneously from more than one computer running the CX-Programmer as long as different tasks are edited.



The cycle time will be increased by from one to several cycle times if the program in the CPU Unit is edited online in MONITOR mode.

The cycle time will also be increased to back up data in the flash memory after online editing. The BKUP indicator will be lit during this period. The progress of the backup is displayed on the CX-Programmer. The delay in the cycle time per cycle will be roughly 1 ms.

There is a limit to the number of edits that can be made consecutively without waiting for backup to be completed. The actual number depends on the type of editing that is performed, but a guideline of 400 edits can be used.

A message will be displayed on the CX-Programmer if the limit is exceeded, and further editing will not be possible until the CPU Unit has completed backing up the data. Once backup has been completed, you can continue online editing again.

Precautions for Correct Use

The cycle time will be longer than normal when a program is overwritten using Online Editing in MONITOR mode, so make sure that the amount of time that it is extended will not exceed the cycle monitoring time set in the PLC Setup. If it does exceed the monitoring time, then a Cycle Time Over error will occur, and the CPU Unit will stop. Restart the CPU Unit by selecting PROGRAM mode first before changing to RUN or MONITOR mode.

Note **1** The internal status (differentiation flags) for DIFFERENTIATE UP instructions (DIFU or instructions with @) and DIFFERENTIATE DOWN instructions (DIFD or instructions with %) contained in the program to be edited online will be initialized.

2 If the task being edited online contains a block program, WAIT status created by WAIT(805), TIMW(813), TIMWX(816), CNTW(814), CNTWX(817), TMHW(815), or TMHWX(818) will be cleared by BPPS(811), and the next execution will be from the beginning.

● Online Editing from CX-Programmer

1 Display the program section that will be edited.

2 Select the instructions to be edited.

3 Select ***Program, Online Edit***, and then ***Begin***.

4 Edit the instructions.

5 Select ***Program, Online Edit***, and then ***Send Changes***. The instructions will be checked and, if there are no errors, they will be transferred to the CPU Unit. The instructions in the CPU Unit will be overwritten and cycle time will be increased at this time.

⚠ Caution

Proceed with Online Editing only after verifying that the extended cycle time will not affect operation. Input signals may not be input if the cycle time is too long.

Additional Information

Temporarily Disabling Online Editing

It is possible to disable online editing for a specific time to ensure response characteristics for machine control in that cycle. Disabling online editing can prevent the cycle time from being inadvertently extended by someone performing online editing.

- Settings for Disabling Online Editing
  Either of the two following settings can be used to disable online editing.
  Set the Online Editing Disable Bit Validator (A527.00 to A527.07) to 5A hex.
  Turn ON the Online Editing Disable Bit (A527.09).
- Operation with Online Editing Disabled
  If there is a request from the CX-Programmer for online editing, the system will enter standby status and not perform online editing. The Online Editing Wait Flag (A201.10) will turn ON. Next, online editing will be performed when the user turns OFF the Online Editing Disable Bit (A527.09). If online editing is already on standby, any online editing operations will be ignored.

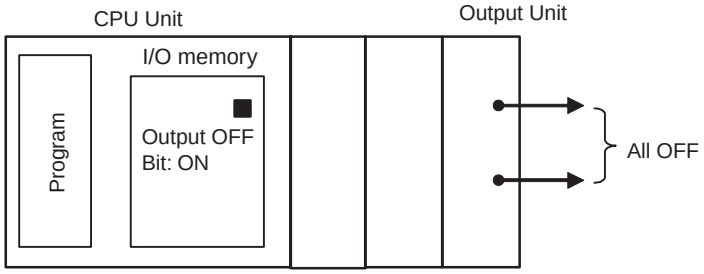
● Related Auxiliary Bits/Words

Name	Address	Description
Online Edit Disable Bit Validator	A527.00 to A527.07	Validates the Online Edit Disable Bit (A527.09). 5A hex: Online Edit Disable Bit valid Not 5A hex: Online Edit Disable Bit invalid
Online Edit Disable Bit	A527.09	To disable online editing, turn this bit ON and set the Online Edit Disable Bit Validator (A527.00 to A527.07) to 5A.
Online Editing Wait Flag	A201.10	ON when an online editing process is on standby because online editing is disabled.

10-7-5 Turning OFF Outputs

Turn ON the Output OFF Bit (A500.15) to turn OFF the outputs of all Output Units regardless of the status of the output bits in I/O memory. This can be used for urgent error processing when the Unit is operating in RUN or MONITOR mode. The INH indicator on the front of the CPU Unit will light yellow.

The status of the Output OFF Bit is maintained even if power is turned OFF and ON if there is a battery.



Additional Information

By default (i.e., if the IOM Hold Bit (A500.12) is turned OFF), all the outputs will be cleared (i.e., turned OFF) if the operating mode is switched from RUN or MONITOR mode to PROGRAM mode. If an output bit in I/O memory is turned ON by the user program, the output will turn ON when I/O is refreshed. To turn OFF all outputs from the Output Units regardless of the operating mode, use the Output OFF Bit.

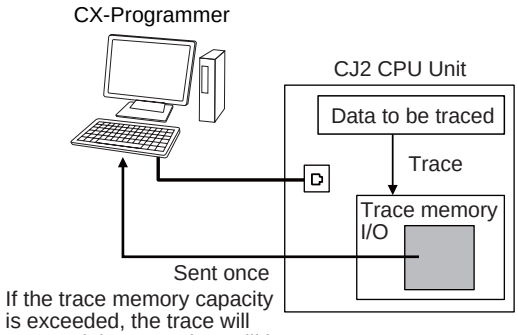
10-7-6 Tracing Data

The Data Trace function samples specified I/O memory data under specified conditions using any one of the following timing methods, and it stores the sampled data in Trace Memory, where they can be read and checked later from the CX-Programmer, as well as saved as files.

The following two trace methods can be used.

● Normal Trace

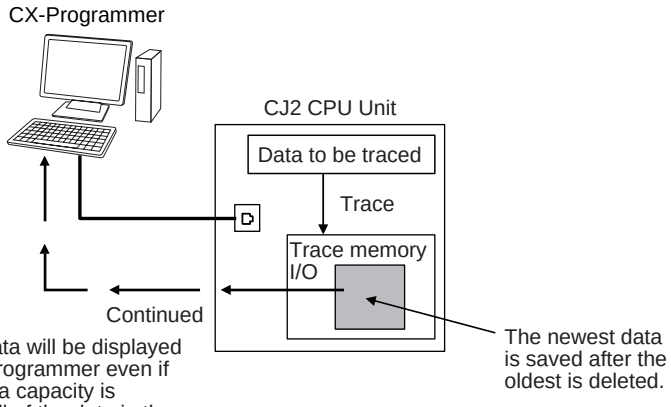
Use this method mainly when high-speed data collection is performed for a short time, such as for debugging. If the trace memory capacity is exceeded, the trace will stop, the trace result currently in trace memory will be displayed, and the data will be stored at the same time in a CSV file.



If the trace memory capacity is exceeded, the trace will stop and the trace data will be sent to the CX-Programmer and displayed.

● Continuous Trace

A continuous trace is used mainly for high-speed data collection over long periods of time. Even if the trace memory capacity is exceeded, the trace will continue, and the trace data will continue to be read to the CX-Programmer. The data will automatically be saved in CSV files whenever the display buffer becomes full. This enables storing long-term trace result data from the start of the trace until the end in multiple CSV files.



The latest data will be displayed on the CX-Programmer even if the trace data capacity is exceeded. All of the data in the display buffer will be automatically saved to a CSV file.

● Data to Sample and Timing

Data Size

The maximum size of I/O memory data that can be specified for sampling is 31 bits and 16 words.

Data Types

The following data types can be specified for sampling.

- BOOL (bit)
- UINT (one-word unsigned binary)
- UDINT (two-word unsigned binary)
- ULINT (four-word unsigned binary)
- INT (one-word signed binary)
- DINT (two-word signed binary)
- LINT (four-word signed binary)
- UINT BCD (one-word unsigned binary)
- UDINT BCD (two-word unsigned binary)
- ULINT BCD (four-word unsigned binary)
- REAL (two-word floating point)
- LREAL (four-word floating point)
- CHANNEL (word)
- WORD (one-word hexadecimal)
- DWORD (two-word hexadecimal)
- LWORD (four-word hexadecimal)

Timing

I/O memory data can be sampled at the following times.

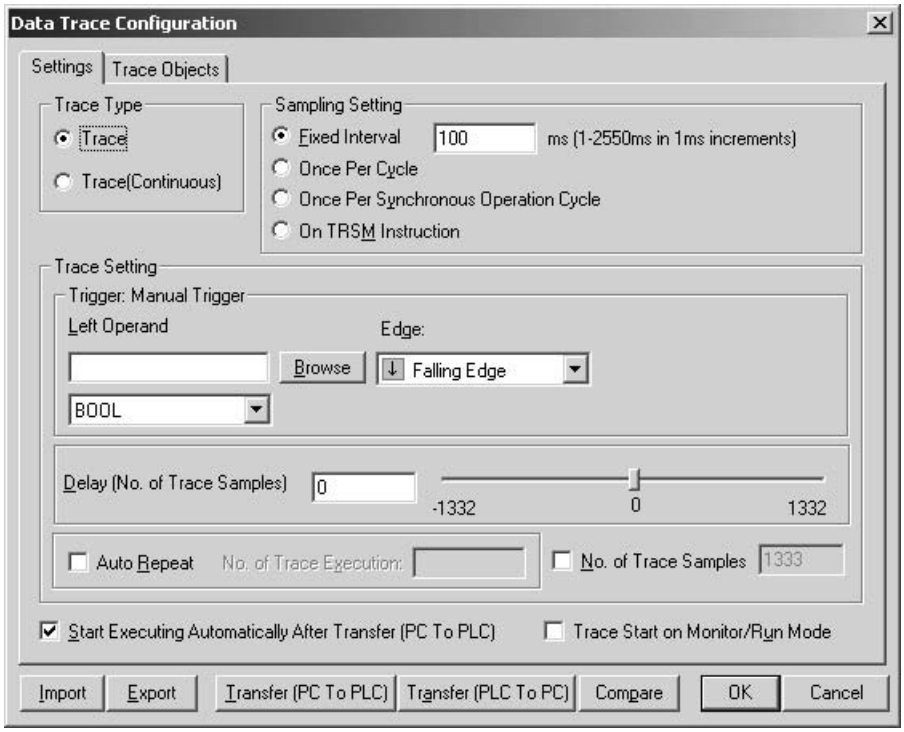
- Specified sampling time (1 to 2,550 ms in 1-ms units)
- Once per cycle
- At the execution of a trace memory instruction (TRSM(045) instruction)
- Once per synchronous operation cycle (0.5 to 10.0 ms in 0.1-ms increments)*

* Continuous tracing cannot be used if the synchronous operation cycle is less than 3.0 ms.

Overview of Data Trace Procedure

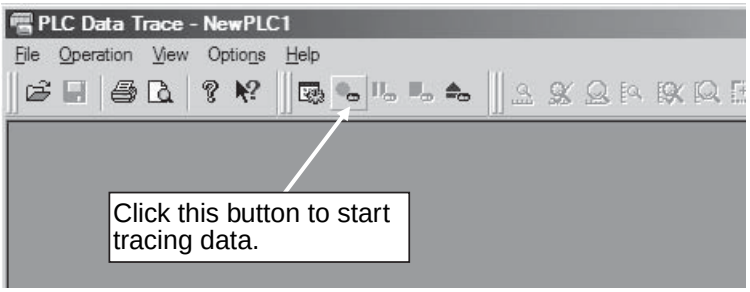
Use the following procedure to execute a trace.

1 Use the CX-Programmer to set trace parameters. (Select ***PLC - Data Trace*** or ***PLC - Time Chart Monitor*** to open the Data Tracing Configuration Window, and then select ***Execute - Set***).

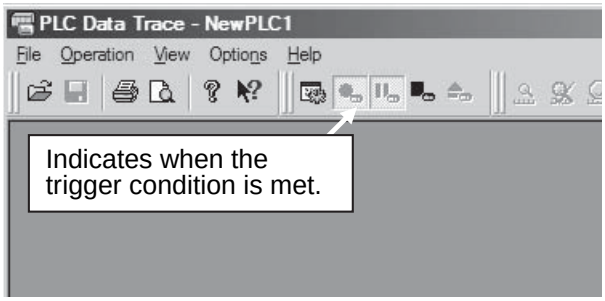


- Executing a Normal Trace
  Select *Trace* in the *Trace Type* Area. Set the address of the data to be sampled on the Trace Objects Tab Page. Multiple addresses can be specified. Make the settings for the trigger condition and delay value in the *Trace Setting* Area on the Settings Tab Page.
- Executing a Continuous Trace
  Select *Trace (continuous)* in the *Trace Type* Area. Set the address of the data to be sampled on the Trace Objects Tab Page. Multiple addresses can be specified. Specify the screen display buffer size, trace time, and folder in which to save the CSV file of collected data in the *Trace Setting* Area on the Settings Tab Page.

2 Turn ON the Trace Start Bit (A508.15) or press the following button. The trace will start.

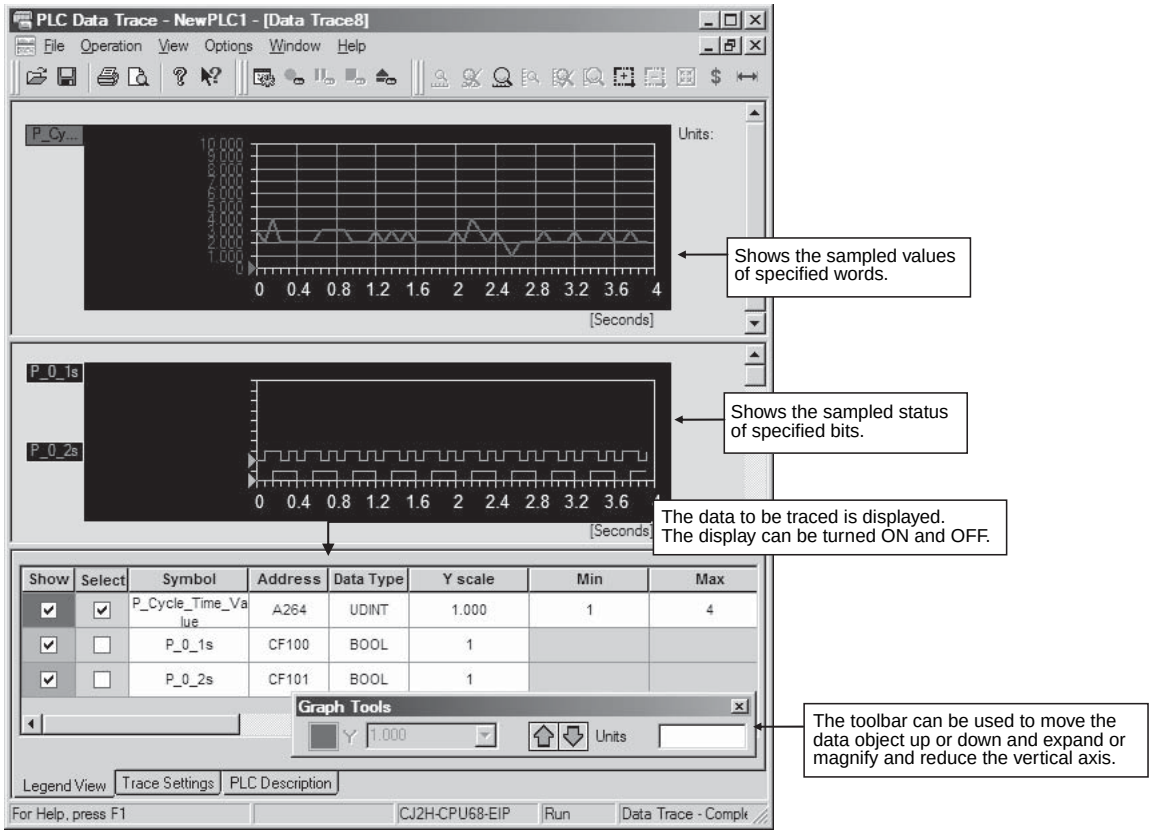


The following display will appear when the trigger conditions have been satisfied.



Data Trace Window

The following figure gives an overview of the Data Trace Window.



For details on the procedure and settings, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).

● CJ2 CPU Unit Data Trace Specifications

The following table compares the data trace specifications of the CJ2 CPU Units and CJ1 CPU Units.

Item	CJ2 CPU Unit	CJ2M CPU Unit	CJ1 CPU Unit
Memory capacity	CPU64(-EIP)/65(-EIP): 8,000 words CPU66(-EIP): 16,000 words CPU67(-EIP)/68(-EIP): 32,000 words Part of the EM Area can be used instead of the data trace area. (Maximum EM Area trace data capacity: 32 Kwords × 25 banks).	8,000 words Part of the EM Area can be used instead of the data trace area (up to 32K words each in four banks).	4,000 words
Sampling size	Sampling bits: 31 Sampling words: 16 One-word data = 16 max. Two-word data = 8 max. Four-word data = 4 max.	Sampling bits: 31 Sampling words: 16 max. One-word data = 16 max. Two-word data = 8 max. Four-word data = 4 max.	Sampling bits: 31 Sampling words: 6 max.
Cycle (when constant)	1 to 2,550 ms (1-ms increments)	1 to 2,550 ms (1-ms increments)	10 to 2,550 ms (10-ms increments)
Automatic start at startup	Possible	Possible	Not possible.
Continuous tracing	Possible It is possible to continue the trace by reading the trace data to the CX-Programmer before exceeding the trace memory capacity.	Possible It is possible to continue the trace by reading the trace data to the CX-Programmer before exceeding the trace memory capacity.	Not possible.
Data length for setting trigger length	Data lengths of 1 word, 2 words, or 4 words	Data lengths of 1 word, 2 words, or 4 words	1 word
Trigger conditions	=, >, ≥, <, ≤, and ≠ can be used.	=, >, ≥, <, ≤, and ≠ can be used.	Only = can be used.
Delay value	–32,768 to 32,767	–32,768 to 32,767	–1,999 to 2,000

10-7-7 Storing the Stop Position at Errors

The Error Flag (P_ER) will turn ON if instructions in the ladder program are outside the range for input parameters. This flag can be referenced to check instruction errors when instructions are executed.

If the *Stop CPU on Instruction Error* Check Box is selected in PLC Setup, a program error will occur if one of the following instruction errors occurs, and operation will be stopped. This function can be used to check if an instruction error has occurred.

● Instruction Errors That Cause Program Errors (A295)

Name	Address	Description
Instruction Processing Error Flag	A295.08	This flag and the Error Flag (P_ER) will be turned ON when an instruction processing error has occurred and the PLC Setup has been set to stop operation for an instruction error.
Indirect DM/EM BCD Error Flag	A295.09	This flag will be turned ON when an indirect DM/EM BCD error has occurred and the PLC Setup has been set to stop operation an indirect DM/EM BCD error.
Illegal Access Error Flag	A295.10	This flag will be turned ON when an illegal access error has occurred (e.g., system area, R/W access to area converted to EM file, write access to write-protected area, or indirect DM/EM BCD error).

The stop position will be stored as described below when operation stops due to a program error.

● Program Error Task (A294)

Name	Address	Description
Program Error Task	A294	Provides the type and number of the task that was being executed when program execution stops as a result of a program error. 0000 to 007F hex (corresponding to tasks 0 to 127) 8000 to 80FF hex (corresponding to interrupt tasks 0 to 255, including extra cyclic tasks)

It is possible to check what task the fatal error occurred in. When a fatal error is cleared, the Program Error Task will be cleared.

● Program Error Position (A299: Upper Bytes, A298: Lower Bytes)

If execution is stopped due to a program error, the following program address for the stop position will be stored.

Note The program address is valid only for ladder programs. It cannot be used with ST or SFC language programs.

- Error in Ladder Program:
  Address counting from the start of the ladder program
- Error in Ladder Action Program:
  Address counting from the start of the ladder action program
- Error in Transition Program:
  Address counting from the start of the transition program
- Error in Function Block
  Address counting from the start of the function block program

10-7-8 Failure Alarm Instructions

The FAL(006) and FALS(007) instructions generate user-defined errors. FAL(006) generates a non-fatal error and FALS(007) generates a fatal error that stops program execution.

When the user-defined error conditions (execution conditions for FAL(006) or FAL(007)) are met, the Failure Alarm instruction will be executed and the following processing will be performed.

- The FAL Error Flag (A402.15) or FALS Error Flag (A401.06) is turned ON.
- The corresponding error code is written to A400 in the Auxiliary Area.
- The error code and time of occurrence are stored in the Error Log.
- The error indicator on the front of the CPU Unit will flash or light.

If FAL(006) has been executed, the CPU Unit will continue operating.
If FALS(007) has been executed, the CPU Unit will stop operating. (Program execution will stop.)

Operation of FAL(006)



When execution condition A goes ON, an error with FAL number 002 is generated, A402.15 (FAL Error Flag) is turned ON, and A360.02 (FAL Number 002 Flag) is turned ON. Program execution continues.

Errors generated by FAL(006) can be cleared by executing FAL(006) with FAL number 00 or performing the error read/clear operation from the CX-Programmer.

Operation of FALS(007)



When execution condition B goes ON, an error with FALS number 003 is generated, and A401.06 (FALS Error Flag) is turned ON. Program execution is stopped.

Errors generated by FAL(006) can be cleared by eliminating the cause of the error and performing the error read/clear operation from the CX-Programmer.

10-7-9 Simulating System Errors

FAL(006) and FALS(007) can be used to intentionally create fatal and non-fatal system errors. This can be used in system debugging to test display messages on Programmable Terminals (PTs) or other operator interfaces.

Use the following procedure.

1 Set the FAL or FALS number to use for simulation in A529. (A529 is used when simulating errors for FAL(006) and FALS(007).)

2 Set the FAL or FALS number to use for simulation as the first operand of FAL(006) or FALS(007).

3 Set the error code and error to be simulated as the second operands (S and S+1) of FAL(006) or FALS(007). Indicate a non-fatal error for FAL(006) and a fatal error for FALS(007).

To simulate more than one system error, specify the same value at A529 for the first operand, and use more than one FAL(006) or FALS(007) instruction with a different second operand.

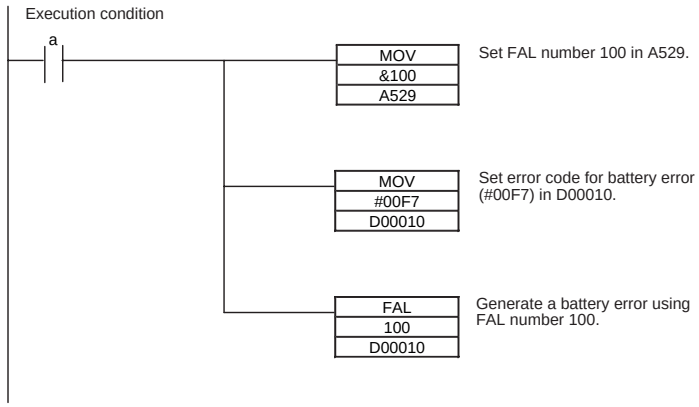
Precautions for Correct Use

This function is used to check if detection is operating correctly in applications by simulating an error to be detected in the CPU Unit. When the system is in operation, disable this function by deleting the FAL(006) or FALS(007) instruction or by always inputting an OFF (P_Off) signal as the input condition.

● Auxiliary Area Flags and Words

Name	Address	Operation
FAL/FALS Number for System Error Simulation	A529	Set a dummy FAL/FALS number to use to simulate the system error. 0001 to 01FF hex: FAL/FALS numbers 1 to 511 0000 or 0200 to FFFF hex: No FAL/FALS number for system error simulation.

● Example for a Battery Error



Precautions for Correct Use

To clear fatal and non-fatal system errors simulated by executing FAL(006) and FALS(007) instructions, use the same methods as for actual system errors. For information on how to clear errors, refer to *Section 6 Troubleshooting* of *CJ2 CPU Unit Hardware User's Manual* (Cat. No. W472). All system errors simulated with FAL(006) and FALS(007) can be cleared by cycling the power supply.

10-7-10 Failure Point Detection

The FPD(269) instruction performs time monitoring and logic diagnosis. The time monitoring function generates a non-fatal error if the diagnostic output isn't turned ON within the specified monitoring time. The logic diagnosis function indicates which input is preventing the diagnostic output from being turned ON.

Time Monitoring Function

FPD(269) starts timing when it is executed and turns ON the Carry Flag if the diagnostic output isn't turned ON within the specified monitoring time. The Carry Flag can be programmed as the execution condition for an error processing block. Also, FPD(269) can be programmed to generate a non-fatal FAL error with the desired FAL number.

When an FAL error is generated, a preset message will be registered and can be displayed on the CX-Programmer. FPD(269) can be set to output the results of logic diagnosis (the address of the bit preventing the diagnostic output from being turned ON) just before the message.

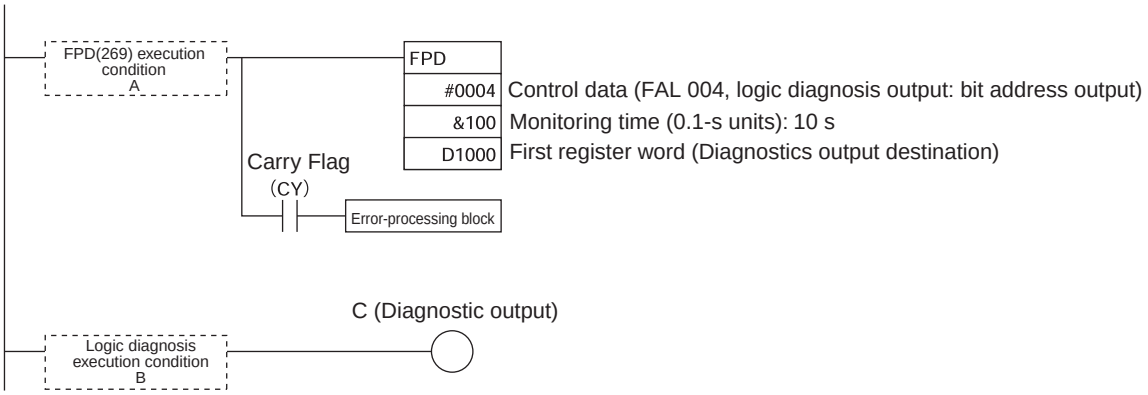
The teaching function can be used to automatically determine the actual time required for the diagnostic output to go ON and set the monitoring time.

Logic Diagnosis Function

FPD(269) determines which input bit is causing the diagnostic output to remain OFF and outputs that bit's address. The output can be set to bit address output (PLC memory address) or message output (ASCII).

If bit address output is selected, the PLC memory address of the bit can be transferred to an Index Register and the Index Register can be indirectly addressed in later processing.

If the message output is selected, the bit address will be registered in an ASCII message that can be displayed on the CX-Programmer.



Time Monitoring:

Monitors whether output C goes ON within 10 seconds after input A. If C doesn't go ON within 10 seconds, a failure is detected and the Carry Flag is turned ON. The Carry Flag executes the error-processing block. Also, an FAL error (non-fatal error) with FAL number 004 is generated.

Logic Diagnosis:

FPD(269) determines which input bit in block B is preventing output C from going ON. That bit address is output to D1000.

Auxiliary Area Flags and Words

Name	Address	Operation
Error Code	A400	When an error occurs, its error code is stored in A400.
FAL Error Flag	A402.15	ON when FAL(006) is executed.
FALS Error Flag	A401.06	ON when FALS(007) is executed.
Executed FAL Number Flags	A360 to A391	The corresponding flag is turned ON when an FAL(006) or FALS(007) error occurs.
Error Log Area	A100 to A199	The Error Log Area contains information on the most recent 20 errors.
Error Log Pointer	A300	When an error occurs, the Error Log Pointer is incremented by 1 to indicate where the next error record will be recorded as an offset from the beginning of the Error Log Area (A100).
Error Log Pointer Reset Bit	A500.14	Turn this bit ON to reset the Error Log Pointer (A300) to 00.
FPD Teaching Bit	A598.00	Turn this bit ON when you want the monitoring time to be set automatically when FPD(269) is executed.

10-8 Synchronous Unit Operation

10-8-1 Overview

The Synchronous Unit Operation Function

The synchronous unit operation function uses a synchronous signal that is generated by the CPU Unit as a specified cycle to synchronize the start of processing between the CPU Units and several Synchronous Units and to synchronize data exchange between these Units. Synchronous Units are CPU Bus Units and Special I/O Units that support synchronous unit operation.

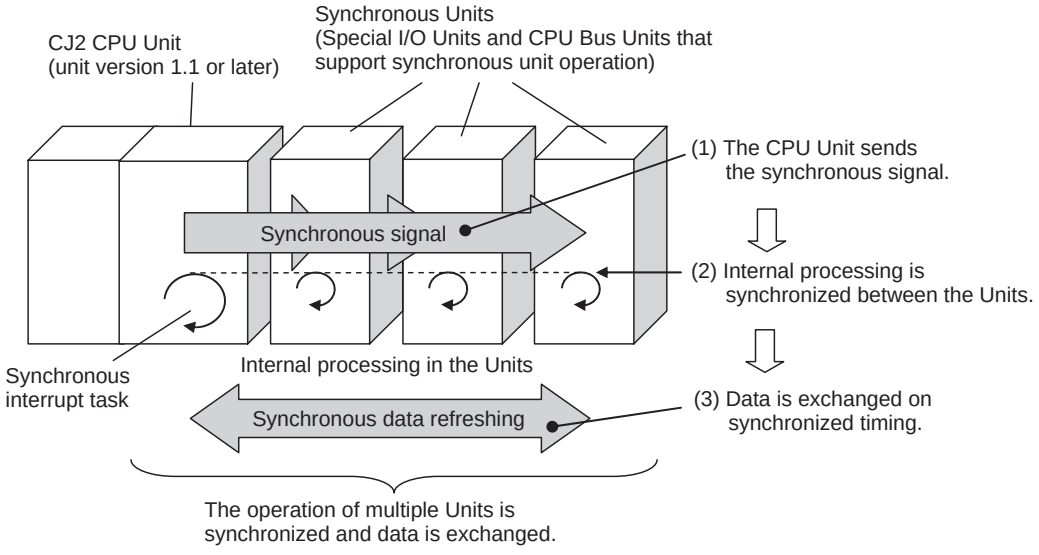
This function is supported by CJ2H CPU Units with unit version 1.1 or later.

Internal Process Synchronization

The CPU Unit can execute an interrupt task in synchronization with data exchange with Synchronous Units. The Synchronous Units can use the synchronous signal from the CPU Unit to synchronize the start of internal processing between all of the Units.

Data Exchange Synchronization

Data can be exchanged between the CPU Unit and multiple Synchronous Units or between Synchronous Units when the synchronous signal is sent.



Precautions for Correct Use

The following requirements must all be met to use the synchronous unit operation function.

- The CJ2H-CPU6□(-EIP) CPU Unit must be unit version 1.1 or later.
- The Special I/O Units and CPU Bus Units must support synchronous unit operation. (These are called Synchronous Units.)

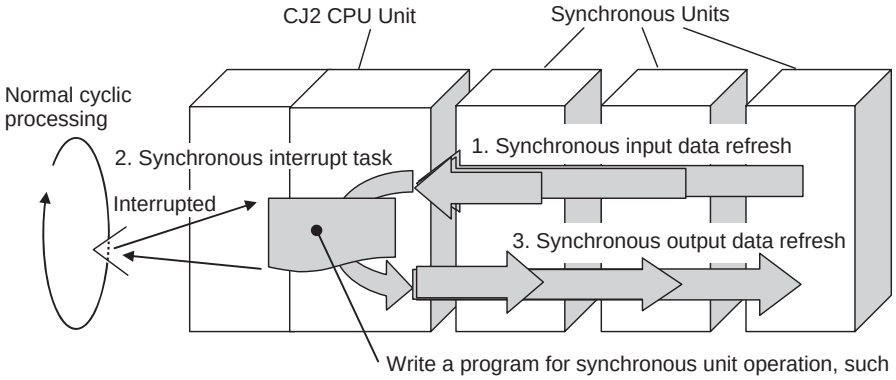
Overall Flow of Synchronous Unit Operation

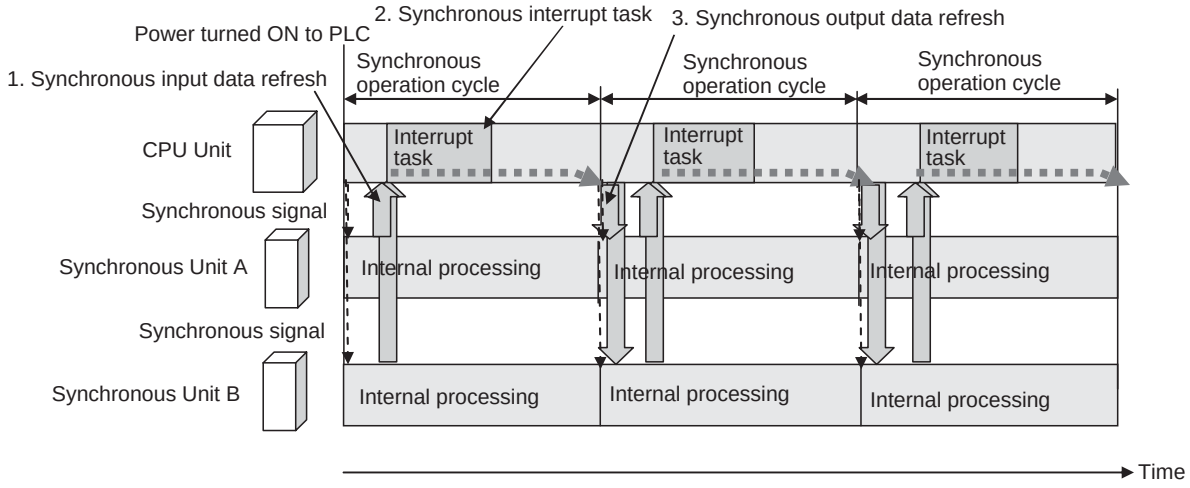
Synchronous unit operation is performed using the following flow.

1 The Synchronous Units synchronize internal processing with each other and send synchronous input data to the CPU Unit once each synchronous operation cycle.

2 The CPU Unit stores the synchronous data in memory and executes an interrupt task for synchronous unit operation. This interrupt task is called the synchronous interrupt task.

3 The CPU Unit sends the results of the synchronous interrupt task to all of the Synchronous Units as synchronous output data.





Item		Description
Synchronous data refresh		Synchronous data is refreshed between the CPU Unit and Synchronous Units on the synchronous operation cycle, which is not affected by the normal cycle time of the CPU Unit. A special area called the Synchronous Data Refresh Area is allocated for synchronous data refreshing. The data in this area is exchanged between the Units as synchronous data.
	Synchronous input data refresh	The input data for the synchronous data refresh is sent from the Synchronous Units to the CPU Unit.
	Synchronous output data refresh	The output data for the synchronous data refresh is sent from the CPU Unit to the Synchronous Units.
Synchronous operation cycle		The synchronous operation cycle is used to generate the synchronous signal, which is used to synchronize the timing of internal processing in the Synchronous Units. This cycle is also used to refresh synchronous data.

Item	Description
Synchronous interrupt task	The synchronous interrupt task is executed after the input data for the synchronous data refresh is received by the CPU Unit. Interrupt task 2 (scheduled interrupt 0) is used for the synchronous interrupt task. Synchronous unit operation can be used without using the synchronous interrupt task.

Overview of Settings for Synchronous Unit Operation

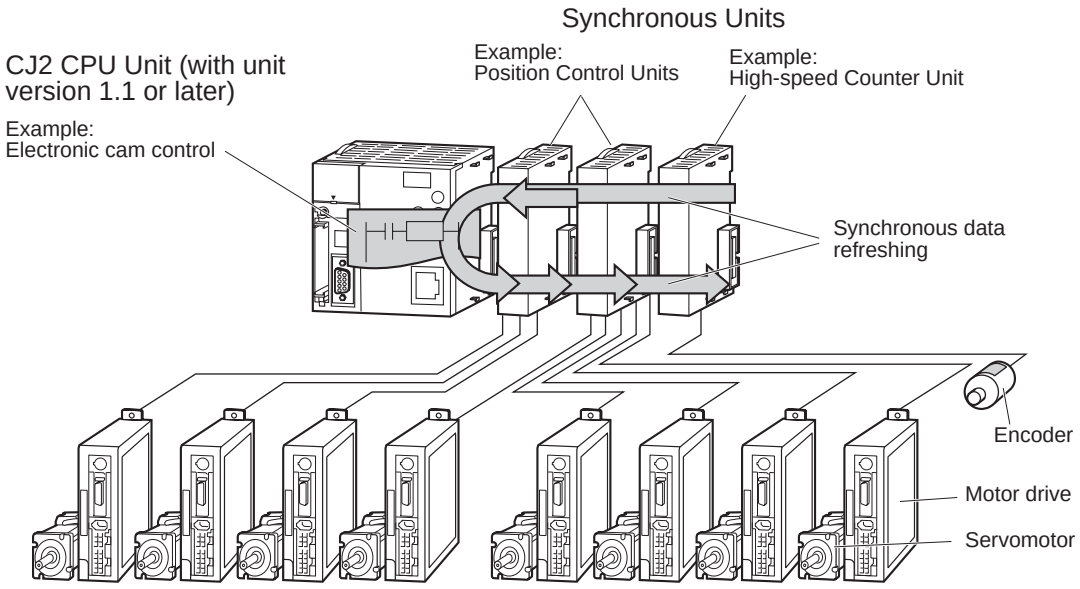
The following settings are used for synchronous unit operation.

Item	Description	Setting method
Synchronous unit operation	You can enable and disable synchronous unit operation.	PLC Setup
Synchronous data refresh	• The Synchronous Units to be used in synchronous unit operation must be registered. • The Synchronous Data Refresh Area must be allocated.	PLC Setup
	The synchronous data for each Synchronous Unit must be set.	The synchronous data settings for each Synchronous Unit is accessed from the PLC Setup
Synchronous operation cycle	The synchronous cycle time is set.	PLC Setup
Synchronous interrupt task	The program in this task is executed.	Select interrupt task 2 (scheduled interrupt 0) in the program properties. Synchronous unit operation can be used without using the synchronous interrupt task.

Application Example

● Synchronizing Operation between Servomotors

In this application, the operation of some servomotors is synchronized according to the operation of an encoder. Here, synchronous data input to the CPU Unit from a High-speed Counter Unit is processed in a synchronous operation program in the synchronous interrupt task, e.g., a program for a digital cam. The results are output from the CPU Unit to some Position Control Units as synchronous data to use in position control.



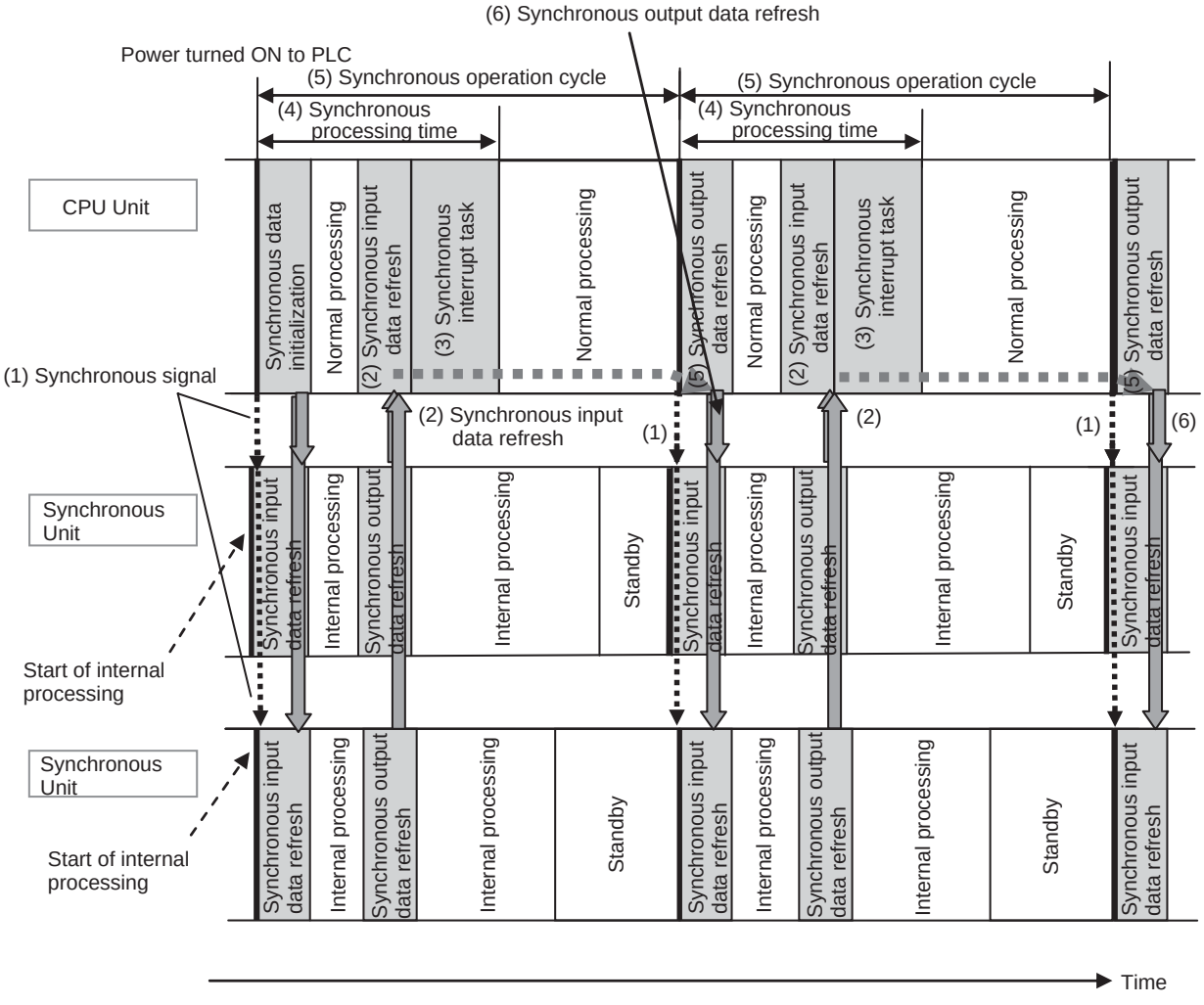
Additional Information

Normal I/O refreshing is also performed between the CPU Unit and Synchronous Units. The differences between the normal I/O refresh and the synchronous data refresh as listed in the following table.

Item	Synchronous data refresh	Normal I/O refresh
Data exchange timing	• Each synchronous operation cycle. (The normal cycle of the CPU Unit is interrupted to perform synchronous data refreshing.) • Data exchange is synchronized with the internal processing in the Synchronous Units.	• Each normal cycle of the CPU Unit (after user program execution). • The timing of the normal I/O refresh is not related to the timing of internal processing in the Synchronous Units.
Data exchange cycle	Constant (always the specified synchronous operation cycle).	Inconsistent (or minimum cycle time if the minimum cycle time is set and the cycle does not exceed the set time).
Start of internal processing in Synchronous Units	Synchronized.	Not synchronized.

10-8-2 Details on Synchronous Unit Operation

The time sequence for synchronous unit operation is shown in the following diagram.



1 Synchronous Signal

When the power supply to the PLC is turned ON, the CPU Unit generates a synchronous signal and then starts initializing the synchronous data. The CPU Units and Synchronous Units also start internal processing at this time.

2 Synchronous Input Data Refresh (Synchronous Units to CPU Unit)

The Synchronous Units sent synchronous data to the CPU Unit.

3 Synchronous Interrupt Task

The CPU Unit executed the synchronous interrupt task. (Synchronous unit operation can be used without using the synchronous interrupt task.)

4 Synchronous Processing Time

The synchronous processing time is the time required for one series of synchronous operation processing in one synchronous operation cycle. It is the time from when the synchronous signal is generated until execution of the synchronous interrupt task has been completed. After the synchronous processing time, normal processing is performed until the synchronous control cycle has expired.

The synchronous processing time must be less than the synchronous operation cycle. (The maximum and present values of the synchronous processing time can be monitored in the Synchronous Operation Status Dialog Box of the CX-Programmer.)

5 Synchronous Operation Cycle

When the synchronous operation cycle time has expired, the synchronous signal is generated again.

6 Synchronous Output Data Refresh (CPU Unit to Synchronous Units)

The CPU Unit sends synchronous data to the Synchronous Units.

7 The overall process is repeated from step 2.

Precautions for Correct Use

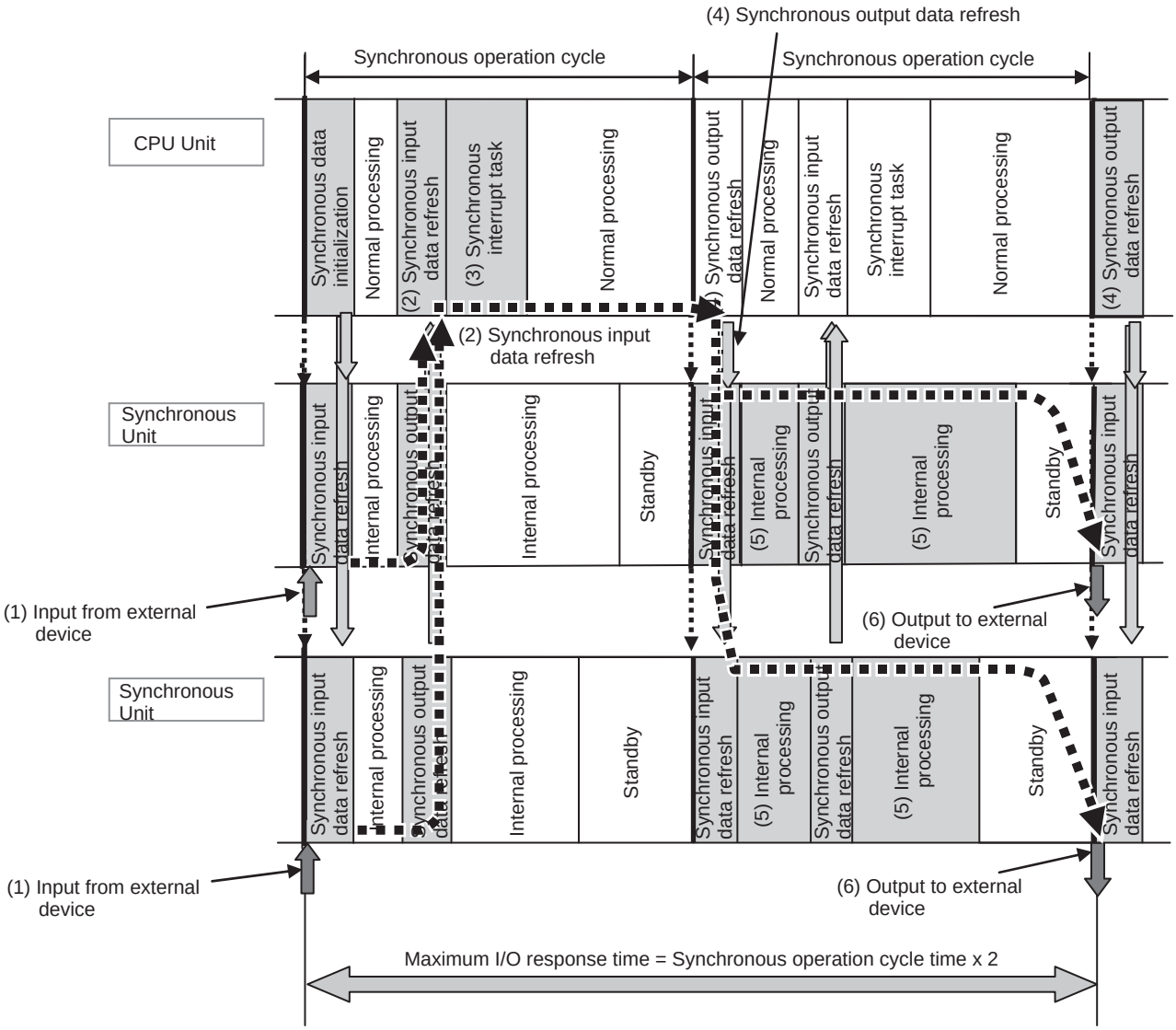
The CPU Unit interrupts normal processing during the synchronous input data refresh, synchronous interrupt task execution, and synchronous output data refresh. The normal cycle time of the CPU Unit will be extended by the time required for these processes. Be sure that the control system will not be adversely affected if the cycle time is extended before using synchronous unit operation.

I/O Response Times for Synchronous Units

The I/O response time for a Synchronous Unit is the total of the times required for the following processes.

(1) Inputs from external devices to Synchronous Units
(2) Synchronous input data refresh (Synchronous Units to CPU Unit)
(3) Synchronous interrupt task execution
(4) Synchronous output data refresh (CPU Unit to Synchronous Units)
(5) Internal processing in Synchronous Unit
(6) Output from Synchronous Unit to external device

The maximum I/O response time is thus two times the synchronous operation cycle time.



Example for a CJ1W-NC□□4 Position Control Unit

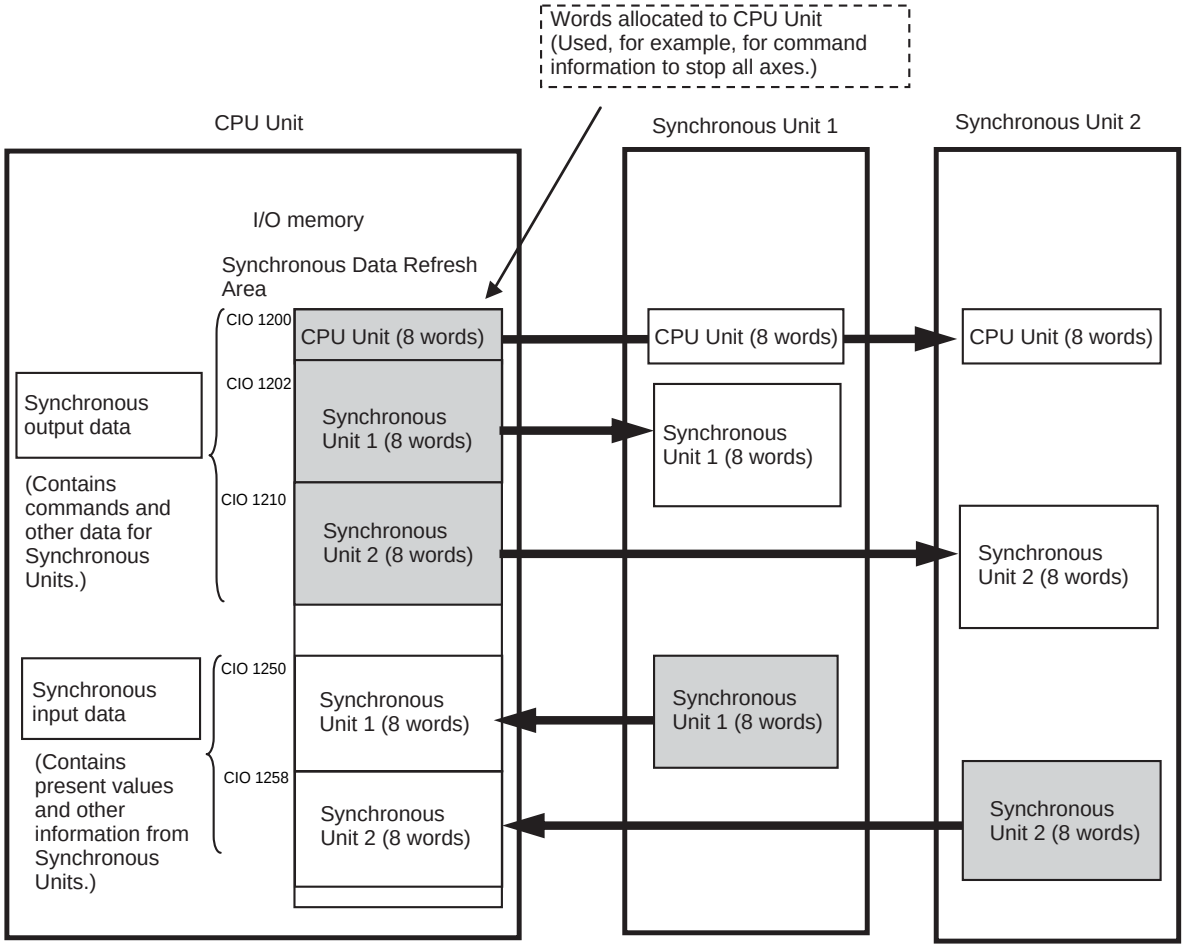
The internal processing time is 1 ms for this Unit, so the synchronous operation cycle time must be at least 1 ms. The I/O response time for this Position Control Unit would thus always be 2 ms.

10-8-3 Synchronous Unit Operation Specifications

Item	Specification
Applicable CPU Units	CJ2H-CPU6□(-EIP) CPU Units with unit version 1.1 or later
Special I/O Units and CPU Bus Units that can be used as Synchronous Units	CJ1W-NC□□4 Position Control Units
Number of mountable Synchronous Units	10 Units max. 5 Units max. for CJ1W-NC□□4 Position Control Units
Mounting location for Synchronous Units	Synchronous Units must be mounted on the CPU Rack. (They will not function if mounted on an Expansion Rack.)
Synchronous operation cycle time	0.5 to 10 ms (in 0.1-ms increments) Set in the PLC Setup. The following conditions must be met: • The synchronous processing time must be less than the synchronous operation cycle time. • The internal processing time of a Synchronous Unit must be less than the synchronous operation cycle time.
Maximum variation in synchronous operation cycle time	10 μs
I/O memory area for synchronized data exchange in CPU Unit	The Synchronous Data Refresh Area is separated into two sections, one for input data from the Synchronous Units to the CPU Unit and one for output data from the CPU Unit to the Synchronous Units. The Synchronous Data Refresh Area is from CIO 1200 to CIO 1295. The contents depends on the Synchronous Units.
Task number of synchronous interrupt task	Interrupt task number 2 must be used. Synchronous unit operation can be used without using the synchronous interrupt task.

10-8-4 Synchronous Data Refresh

The Synchronous Data Refresh Area in the CIO Area of the CPU Unit always starts at CIO 1200. The output data sent from the CPU Unit to Synchronous Units is first and is followed by the input data sent from the Synchronous Units to the CPU Unit.



The following settings are made in the *Synchronous Unit Operation Settings* Area of the Timings/Synchronous Tab Page in the PLC Setup.

- The total size of the output data and the total size of the input data
- The output size and input size for each Unit
- The start address of input area

Addresses in the Synchronous Data Refresh Area are from CIO 1200 to CIO 1295. The first word in this area is always CIO 1200. Other addresses can be set.

I/O	Item	Description
Output data	Direction	CPU Unit to Synchronous Units
	Starting word	CIO 1200 (fixed)
	Allocation	The sizes of data specified in the PLC Setup are allocated to the CPU Unit and then to the Synchronous Units in the order the Synchronous Units are registered. Two or more words can be allocated to the CPU Unit and zero or more words can be allocated to each Synchronous Unit.
Input data	Direction	Synchronous Units to CPU Unit
	Starting word	The address of the starting word is set in the PLC Setup. It can be between CIO 1202 and CIO 1294.
	Allocation	The sizes of data specified in the PLC Setup are allocated to the Synchronous Units in the order the Synchronous Units are registered. Zero to 16 words can be allocated to each Synchronous Unit.

● Allocation Example

Input Data and Output Data Word Allocation Example for Synchronous Data

The start addresses and data sizes are set for synchronous data refreshing. (This example uses unit numbers 0 and 1.)

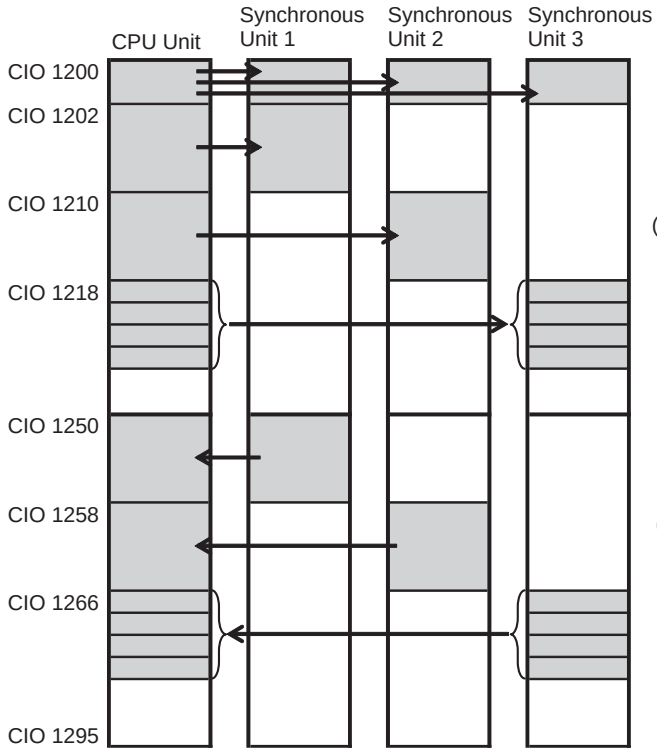
Address	Item	I/O
CIO 1200	Words allocated to the CPU Unit (for a data size setting of 2 words)	Outputs from CPU Unit to Synchronous Units
CIO 1201		
CIO 1202	Words allocated to Synchronous Unit 0 (for a data size setting of 2 words)	
CIO 1203		
CIO 1204	Words allocated to Synchronous Unit 1 (for a data size setting of 4 words)	
CIO 1205		
CIO 1206		
CIO 1207		
:	:	
CIO 1250	Words allocated to Synchronous Unit 0 (for a data size setting of 4 words)	Inputs from Synchronous Units to CPU Unit
CIO 1251		
CIO 1252		
CIO 1253		
CIO 1254	Words allocated to Synchronous Unit 1 (for a data size setting of 2 words)	
CIO 1255		
:	:	
CIO 1295	...	

Setting Example for I/O Data for Synchronous Data Refresh Using Position Control Units

The application of the allocated I/O data must be set.

- Output Data
  For a Position Control Unit, the word that is used for position data for a synchronous feed command is set for each axis.
- Input Data
  For a Position Control Unit, the present command value or present feedback position for each axis is set.

Setting Example



(4) Output Data Assignments for Synchronous Units
(Setting the Destination of Output Data)

CIO 1218: Position Data for Synchronous Feed Command for Axis 1
CIO 1220: Position Data for Synchronous Feed Command for Axis 2
CIO 1222: Position Data for Synchronous Feed Command for Axis 3
CIO 1224: Position Data for Synchronous Feed Command for Axis 4

(5) Input Data Assignments for Synchronous Units
(Setting the Data To Be Input)

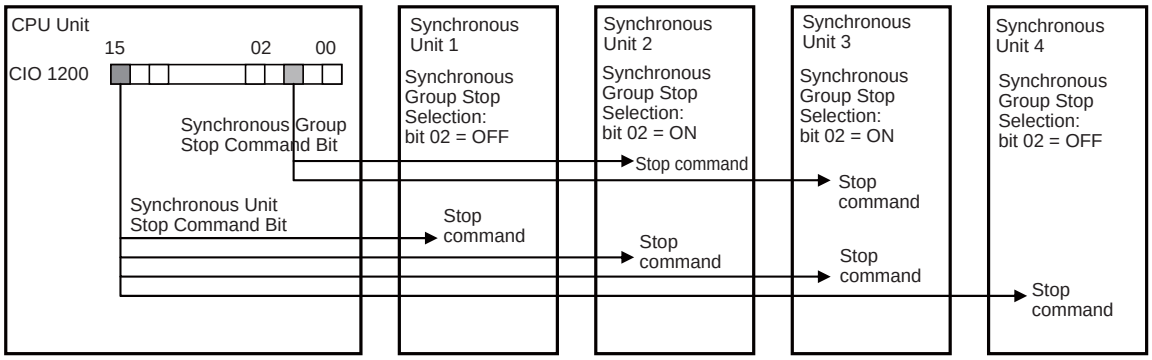
CIO 1266 to CIO 1267: Present Feedback Position for Axis 1
CIO 1268 to CIO 1269: Present Feedback Position for Axis 2
CIO 1270 to CIO 1271: Present Feedback Position for Axis 3
CIO 1272 to CIO 1273: Present Feedback Position for Axis 4

● Application Example

I/O data is used as described in the following table when using Position Control Units and High-speed Counter Units and Synchronous Units.

Data	Application example
Words allocated to CPU Unit	Synchronous Group Stop Command Bits, Synchronous Unit Stop Command Bit,* etc.
Output data for synchronous data refresh	Target position data (X, Y, Z, and U axes)
Input data for synchronous data refresh	Present feedback data (X, Y, Z, and U axes)

* Synchronous Group Stop Command Bits and Synchronous Unit Stop Command Bit
Command bits can be used from the CPU Unit to stop the function that is currently being executed for Synchronous Units that are performing synchronous unit operation. There are two command bits. The Synchronous Unit Stop Command Bit applies to all Synchronous Units. The Synchronous Group Stop Command Bits apply to preset groups of Synchronous Units.



Synchronous Unit Stop Command Bit

Bit 15 in CIO 1200 in the Synchronous Data Refresh Area is the Synchronous Unit Stop Command Bit. If this bit is turned ON, a command will be sent to stop the operation currently being preformed by all Synchronous Units.

Synchronous Group Stop Command Bit

Bits 00 to 14 in CIO 1200 in the Synchronous Data Refresh Area are the Synchronous Group Stop Command Bits. If one of these bits is turned ON, a command will be sent to stop the operation currently being preformed by all Synchronous Units for which the same bit is ON in the Synchronous Group Stop Selection parameter setting the Synchronous Unit.

The Synchronous Group Stop Selection parameter settings in the axis parameters of a Position Control Unit are given in the following table.

Name	Bit	Function	Settings	Default
Synchronous Group Stop Selection	00	These bits specify the operation of the Synchronous Unit for each bit in CIO 1200 in the Synchronous Data Refresh Area. OFF: Ignore bit 00. ON: Stop if bit 00 turns ON.	OFF, ON	OFF
	01	OFF: Ignore bit 01. ON: Stop if bit 01 turns ON.	OFF, ON	OFF
	02	OFF: Ignore bit 02. ON: Stop if bit 02 turns ON.	OFF, ON	OFF
	:	:		
	14	OFF: Ignore bit 14. ON: Stop if bit 14 turns ON.	OFF, ON	OFF
	15	---	---	---

10-8-5 Restrictions in Using Synchronous Unit Operation

● Restrictions on the Synchronous Operation Cycle Time

Set the synchronous operation cycle time so that the following two conditions are met.

1. The synchronous processing time must be less than the synchronous operation cycle time.

* Generally speaking, set the synchronous operation cycle time to 1.5 times the synchronous processing time or higher.

2. The internal processing time of a Synchronous Unit must be less than the synchronous cycle time.

1. Synchronous Processing Time

The synchronous processing time is the total of the following two times.

(1) The time from when the synchronous signal is generated until execution of the synchronous interrupt task has been completed (i.e., the total of the following: synchronous output data refresh time, normal processing time, synchronous input data refresh time, and the synchronous input task execution time)

(2) The time required to executed any I/O interrupt tasks or external interrupt tasks that occur during the processing described in (1), above. (The maximum and present values of the synchronous processing time can be monitored in the Synchronous Operation Status Dialog Box of the CX-Programmer.)

2. Internal Processing Time of a Synchronous Unit

The internal processing time of a CJ1W-NC□□4 Position Control Unit is 1 ms, so the synchronous operation cycle time must be at least 1 ms.

3. Failure to Meet the Above Conditions

Synchronous Operation Cycle Time Is Shorter Than the Synchronous Processing Time

The next synchronous operation cycle will be entered during execution of the synchronous interrupt task, and I/O interrupt task, or external interrupt task. This will cause a synchronous processing time over error in the CPU Unit. The synchronous interrupt task and synchronous data refresh will not be executed once in that synchronous operation cycle.

Synchronous Operation Cycle Time Is Shorter Than the Internal Processing Time of the Synchronous Unit

The next synchronous operation cycle will be entered before the Synchronous Unit completes internal process. This will cause a synchronous processing time over error in the Synchronous Unit. Processing will continue in the Synchronous Unit (e.g., axis operations will continue for a Position Control Unit).

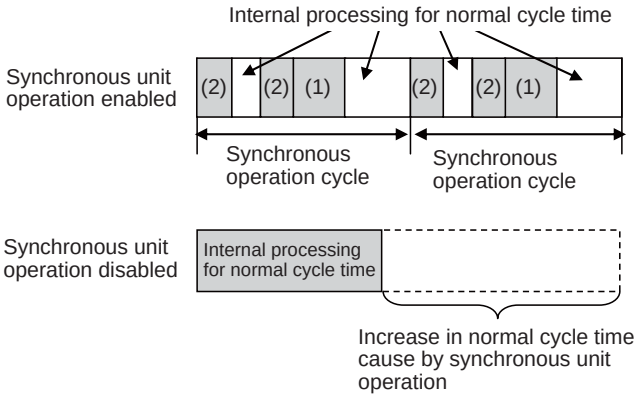
● Increases in the Normal Cycle Time

When synchronous unit operation is used, the normal cycle time will be increased by the following times.

(1) The execution time for the synchronous interrupt time (This is the same as for normal scheduled interrupts.)

(2) The synchronous input and output data refresh times

Note The above times are the same as the synchronous processing time minus the normal processing time performed during that time.



● Scheduled Interrupt Tasks 0 and 1 Disabled

If synchronous unit operation is used, scheduled interrupts 0 and 1 cannot be used to execute interrupt tasks. Thus, scheduled interrupt task 0 (interrupt task 2) and scheduled interrupt task 1 (interrupt task 3) cannot be used. Interrupt task 2, however, is used as the synchronous interrupt task.

● Restrictions for Interrupt Control Instructions

The MSKS, MSKR, and CLI instructions cannot be used in the synchronous interrupt task. The DI and EI instructions can be used.

● Restrictions for Timer Instructions

- The following restrictions apply to the TIM/TIMX, TIMH/TIMHX, and TMHH/TMHHX instructions.
  - The timers will not operate correctly if the cycle time exceeds 100 ms.
  - If one of these timer instructions is used in a task that is stopped or is not executed because it is jumped by a JMP, CJMP, or CJPN instruction, there will be the following error in the timer
    TIM/TIMX: –10 ms
    TIMH/TIMHX: –10 ms
    TMHH/TMHHX: –1 ms
  - The timer PVs cannot be accessed accurately.

● Restrictions on Using Instructions in Interrupt Tasks When High-speed Interrupt Function Is Enabled

If using the synchronous unit operation function is enabled, the high-speed interrupt function will be unconditionally enabled in the PLC Setup. Thus, some instructions, such as network communications instructions, cannot be used in any interrupt tasks, including the synchronous interrupt task. For details, refer to *10-2-6 High-speed Interrupt Function*.

● Other Restrictions When High-speed Interrupt Function Is Enabled

- Background processing cannot be used in interrupt tasks.
- The data in the following Auxiliary Area words will not be valid: A440 (Maximum Interrupt Task Processing Time) and A441 (Interrupt Task with Maximum Processing Time).
- The increase of the cycle time cause by using EtherNet/IP tag data links will be even longer.

● Mounting Location for Synchronous Units

Synchronous Units must be mounted on the CPU Rack. They cannot be mounted on Expansion Racks.

10-8-6 Application Procedure

The procedure to use synchronous unit operation is given below.

1 Initial Settings (Refer to *10-8-7 PLC Setup*.)

Make the synchronous unit operation settings on the Timings/Synchronous Tab Page of the PLC Setup from the CX-Programmer.

(1) Enable synchronous unit operation. Select the *Use Synchronous Operation* Check Box.

(2) Set the synchronous operation cycle time.

(3) Set the words for exchanging data with the synchronous data refresh.

(a) Set the start address for the input data in the *Synchronous Data Refresh Area Allocation* Area.

(b) Register the Synchronous Units.

(c) Set the start address and data size for the input and output data for each Synchronous Unit.

2 I/O Data Assignments for Synchronous Units

Assign how the I/O data will be used for each Synchronous Unit. Click the **Special Unit Setup** Button in the *Synchronous Unit Operation Settings* Area of the Timings/Synchronous Tab Page of the PLC Setup from the CX-Programmer to make the settings for each Synchronous Unit. Refer to the manual for each Synchronous Unit for information on specific settings.

3 Writing the Synchronous Interrupt Task If It Is Required. (Refer to *10-8-8 Writing the Synchronous Interrupt Task*.)

Write the program for the synchronous interrupt task to interrupt task 2 from the CX-Programmer.

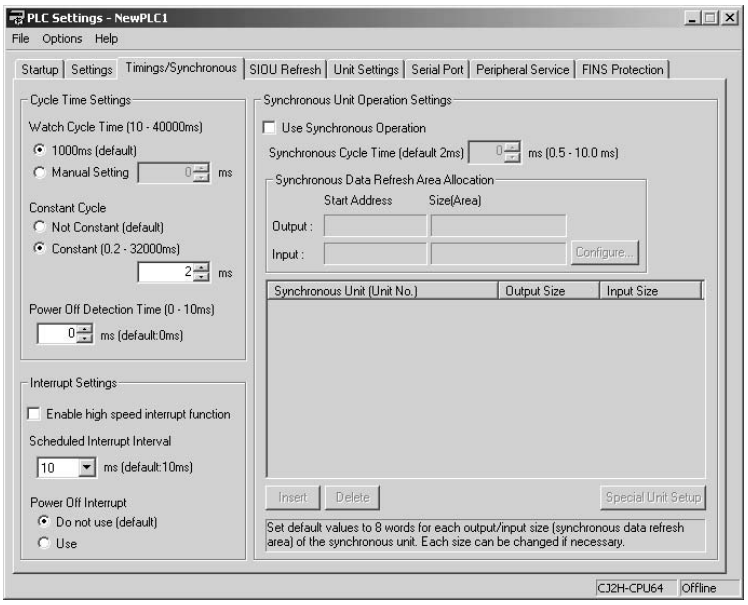
4 Debugging Synchronous Unit Operation (Refer to *10-8-9 Adjusting and Troubleshooting Synchronous Unit Operation*.)

Adjust the synchronous operation cycle time while monitoring the following values on the *Synchronous Operation Status* Dialog Box of the CX-Programmer.

- Present and maximum values of the synchronous processing time
- Set value of the synchronous operation cycle time
- The present and maximum values of the normal cycle time
- Data exchange errors with Synchronous Units

10-8-7 PLC Setup

Settings for the synchronous unit operation function are made in the *Synchronous Unit Operation Settings Area* of the Timings/Synchronous Tab Page from the CX-Programmer.



● Use Synchronous Operation

This check box is used to enable and disable the synchronous unit operation function.

Parameter	Settings	Default	Description	Related Auxiliary Area bits and words
Use Synchronous Operation	Cleared: Not used. Selected: Used.	Cleared (not used)	Select the check box to use synchronous unit operation. If using the synchronous unit operation function is enabled, the high-speed interrupt function will be unconditionally enabled.	---

● Synchronous Cycle Time

Set the synchronous operation cycle time.

Parameter	Settings	Default	Description	Related Auxiliary Area bits and words
Synchronous Cycle Time	0.5 to 10.0 ms (in increments of 0.1 ms)	2.0 ms	Set the synchronous operation cycle time.	A10102 (Synchronous Operation Cycle Time)

Precautions for Correct Use

The following conditions must be met for the synchronous operation cycle time.

- The synchronous processing time must be less than the synchronous operation cycle time.
- The internal processing time of a Synchronous Unit must be less than the synchronous cycle time.

Generally speaking, set the synchronous operation cycle time to 1.5 times the synchronous processing time or higher.

● Synchronous Data Refresh Area Allocation

1 Output and Input Data Settings for Synchronous Units

Set the data sizes for synchronous data refreshing.

Parameter	Settings	Settings	Default	Description	Related Auxiliary Area bits and words
Output	Start Address	CIO 1200 (fixed)	CIO 1200	This is the address of the first word for the synchronous output data refresh.	---
	Size (Allocation Area)	2 to 96 words*	50 words	This is the size for the synchronous output data refresh.	---
Input	Start Address	CIO 1202 to CIO 1294 or “Not used”	CIO 1250	This is the address of the first word for the synchronous input data refresh.	---
	Size (Allocation Area)	2 to 94 words or no setting*	46 words	This is the size for the synchronous input data refresh.	---

* The size is calculated automatically when the start address is set.

2 Size (Area)

Output Data
Set the data size for the synchronous output data refresh. The start address will be calculated automatically.
Set the output size separately for each Synchronous Unit.

Parameter	Settings	Default	Description	Related Auxiliary Area bits and words
Output size	CPU Unit: 2 words or higher (to maximum output size) Synchronous Units: 0 words or higher (to maximum output size)	CPU Unit: 2 words Synchronous Units: 8 words	Set the data size for the synchronous output data refresh.	---

Input Data
Set the data size for the synchronous input data refresh. The start address will be calculated automatically.
Set the output size separately for each Synchronous Unit.

Parameter	Settings	Default	Description	Related Auxiliary Area bits and words
Input size	Synchronous Units: 0 to 16 words	Synchronous Units: 8 words	Set the data size for the synchronous input data refresh.	---

● Synchronous Unit (Unit No.)

Register the Synchronous Units in the PLC that are to be used in synchronous unit operation. Normally up to ten Units can be registered, but fewer Units can be registered depending on the Unit. Refer to the operation manual for each Unit for details.

Parameter	Settings	Default	Description	Related Auxiliary Area bits and words
Synchronous Units	Names of CPU Units and other Units that support synchronous unit operation	CPU Unit only	Click an empty row to display a list of Synchronous Units. Select a Unit from the list.	---

10-8-8 Writing the Synchronous Interrupt Task

Write the program to be executed in the synchronous interrupt task and set the task type to interrupt task 2 (scheduled interrupt 0) in the program properties. This task will include programming for synchronous control, such as for digital cam control. For example, for digital cam control, this task would contain ARITHMETIC PROCESS (APR) instructions or standard function blocks to control a cam curve based on a real or virtual axis.

● Related Auxiliary Area Bits and Words

Name	Bit/word	Description	Read/Write
Synchronous Unit Operation Servicing Flag	A10100.00	This flag turns ON from the second synchronous operation cycle when the CPU Unit can service Synchronous Units for the synchronous unit operation function. It is OFF at all other times, including the first synchronous operation cycle after starting or restarting the PLC. Use this flag in user programming to control program execution related to synchronous unit operation.	Read
Synchronous Input Data Refresh Error Code	A10101	This word contains 0001 hex when the CPU Unit fails to receive synchronous input data from a Synchronous Unit. It contains 0000 hex after synchronous input data is successfully received. It is updated each synchronous operation cycle. Use this word in user programming as a condition for using synchronous input data.	Read
Synchronous Operation Cycle Time	A10102	This word contains the synchronous operation cycle time set in the PLC Setup in 0.1-ms increments from the point when the set cycle time is valid. Use this word to read the set synchronous operation cycle time from user programming, such as from a function block.	Read

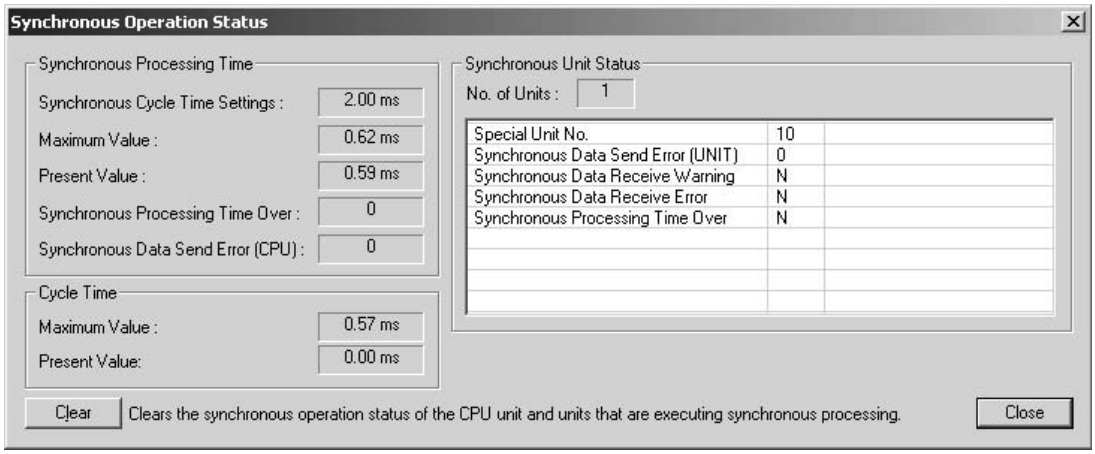
10-8-9 Adjusting and Troubleshooting Synchronous Unit Operation

Checking Synchronous Unit Operation Status

The status of the synchronous unit operation can be checked from the Synchronous Operation Status Dialog Box in the CX-Programmer or from the SYNC indicators on the Synchronous Units.

● CX-Programmer

Place the CX-Programmer online with a PLC that is using synchronous unit operation and select ***PLC - Edit - Synchronous Operation Status*** from the PLC Information Menu. The Synchronous Operation Status Dialog Box shown below will be displayed.



Item		Description
Synchronous Processing Time	Synchronous Cycle Time Settings	The value that is set for the synchronous operation cycle.
	Maximum Value and Present Value	The maximum synchronous operation cycle time to the present and the present synchronous operation cycle time.
	Synchronous Processing Time Over	The number of times that the synchronous processing time of the CPU Unit has exceeded the set synchronous operation cycle time. This value is cleared when the PLC is started.
	Synchronous Data Send Error (CPU)	The number of times that the CPU Unit failed to send synchronous data to the Synchronous Units. This value is cleared when the PLC is started.
Cycle Time	Maximum Value and Present Value	The maximum and present values of the normal cycle time.

Item		Description
Synchronous Unit Status	No. of Units	The number of Units that are set for synchronous unit operation.
	Special Unit No.	The unit number of the Synchronous Unit as a Special I/O Unit or CPU Bus Unit.
	Synchronous Data Send Error (UNIT)	The number of times that CPU Unit failed to receive data sent by a Synchronous Unit. This value is cleared when the PLC is started.
	Synchronous Data Receive Warning	This status will be *Yes* if the Synchronous Unit has failed to receive synchronous data sent from the CPU Unit even one time (including if the CPU Unit failed to send the synchronous output data). This status will be cleared if the Unit Warning Reset Bit in the CPU Bus Unit Area or Special I/O Unit Area is turned ON.
	Synchronous Data Receive Error	This status will be *Yes* if the Synchronous Unit has failed to receive synchronous data sent from the CPU Unit two or more times in a row. This status will be cleared if the Unit Error Reset Bit in the CPU Bus Unit Area or Special I/O Unit Area is turned ON.
	Synchronous Processing Time Over	This status will be *Yes* if the internal processing time in the Synchronous Unit has exceeded the synchronous operation cycle time one or more times. This status will be cleared if the Unit Warning Reset Bit in the CPU Bus Unit Area or Special I/O Unit Area is turned ON.
Clear Button		This button clears the synchronous unit operation status and error/warning values both in the CPU Unit and the Synchronous Units.

Adjusting the Synchronous Operation Cycle Time

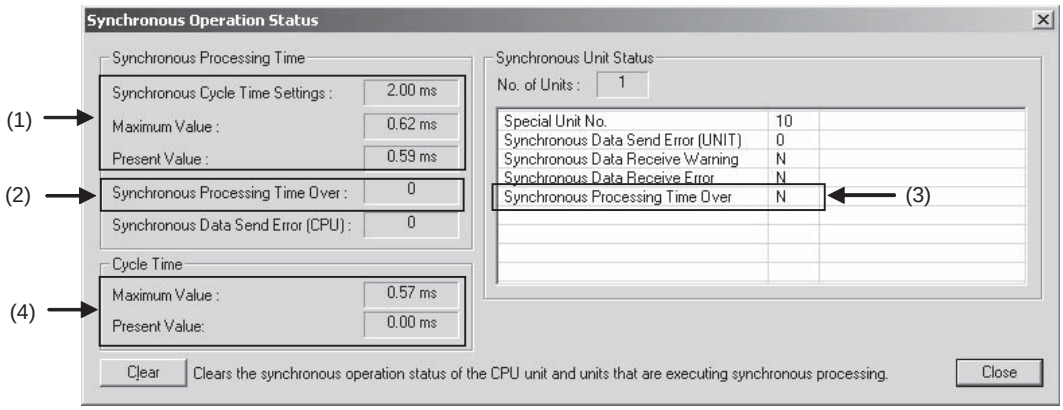
You can use the following procedure to check to see if the set synchronous operation cycle time is suitable and if necessary adjust it.

(1) Check to see if the maximum and present values of the synchronous operation cycle exceed the set value.

(2) Check to see if a Synchronous Processing Time Over Error is occurring frequently in the CPU Unit.

(3) Check to see if a Synchronous Processing Time Over Warning is occurring frequently in a Synchronous Unit.

(4) Check the maximum and present values of the normal cycle time to see if the normal cycle time has exceeded the allowable range due to the increase caused by synchronous unit operation.



● Checking the SYNC Indicators on the Fronts of Synchronous Units

The SYNC indicator on the front of a Synchronous Unit will be lit green when the Unit is in Synchronous Unit Operation Mode. This indicator will go out for synchronous output data reception errors and other errors.



Additional Information

The following bits are allocated to the Synchronous Unit in the Special I/O Unit Area or CPU Bus Unit Area. This bits can be used in the user programming in the CPU Unit to check and control the status of the Synchronous Units.

I/O classification	Size	Name	Function
Input bits allocated in the Special I/O Unit Area or CPU Bus Unit Area (input to CPU Unit)	1 bit	Synchronous Unit Operation Mode Flag	This flag will be ON while the Synchronous Unit is operating in Synchronous Unit Operation Mode. (This flag has the same status as the SYNC indicator on the Synchronous Unit.)
	1 bit	Synchronous Data Receive Warning Flag	This flag will be ON if the Synchronous Unit has failed to receive synchronous data sent from the CPU Unit even one time.
	1 bit	Synchronous Data Receive Error Flag	This flag will be ON if the Synchronous Unit has failed to receive synchronous data sent from the CPU Unit two or more times in a row.
	1 bit	Synchronous Processing Time Over Flag	This flag will be ON if the internal processing time in the Synchronous Unit has exceeded the synchronous operation cycle time one or more times.
Output bits allocated in the Special I/O Unit Area or CPU Bus Unit Area (output from CPU Unit)	1 bit	Unit Error Reset Bit	Turn ON this bit to reset errors, such as synchronous data reception errors, in the Synchronous Unit.
	1 bit	Unit Warning Reset Bit	Turn ON this bit to reset warnings, such as synchronous data reception warnings and synchronous processing time exceeded warnings, in the Synchronous Unit.

Stoppage of Synchronous Unit Operation

Synchronous Signal Monitoring Errors

If the synchronous signal from the CPU Unit to Synchronous Unit is not generated for over 100 ms during normal operation and 11 s at startup, all Synchronous Units and all synchronous unit operation will be stopped.

I/O Bus or Synchronous Unit Errors

If even one Synchronous Unit has an I/O bus error or any of the following Unit errors, all Synchronous Units and all synchronous unit operation will be stopped.

- I/O setting error
- Duplicate unit number error
- Special I/O Unit setting error
- CPU Bus Unit setting error
- Special I/O Unit error
- CPU Bus Unit error

A dialog box that shows the error will be displayed on the CX-Programmer.



Additional Information

If even one of the Synchronous Unit is restarted during synchronous unit operation, the synchronous operation cycle will be stopped and all Synchronous Units will be restarted. Synchronous unit operation will be restarted only after all Synchronous Units have restarted normally.

Adjustments and Troubleshooting

Adjustments, change as changing the synchronous operation cycle time, are necessary for the problems described in the following table.

Synchronous operation status on CX-Programmer	Problem	Cause	Remedy
Synchronous Processing Time Area	A Synchronous Processing Time Over Error occurs frequently.	Synchronous operation processing (synchronous data refreshing plus synchronous interrupt task execution) could not be completed within the synchronous operation cycle.	Set the synchronous operation cycle time longer than the synchronous operation processing time (synchronous data refresh times plus synchronous interrupt task execution time).
	A Synchronous Data Send Error (CPU) occurs.	A hardware error occurred in the CPU Unit.	Replace the CPU Unit.
Synchronous Unit Status Area	A Synchronous Data Send Error (UNIT) occurs frequently.	The synchronous input data set from the Synchronous Unit to the CPU Unit could not be prepared before the synchronous input data refresh period.	Set the synchronous operation cycle time to longer than the internal processing time of the Synchronous Unit.
	A Synchronous Processing Time Over Warning occurs	Internal processing in the Synchronous Unit could not be completed before the end of the synchronous operation cycle.	Set the synchronous operation cycle time to longer than the internal processing time of the Synchronous Unit.

11

Programming Devices and Communications

This section describes how to access the PLC from the CX-Programmer. It also describes serial communications and communications across networks.

11-1 Accessing a PLC from the CX-Programmer 11-2
11-1-1 Overview 11-2
11-1-2 System Configurations for Accessible PLCs 11-4
11-1-3 Accessing a PLC from the CX-Programmer 11-8
11-1-4 Automatic Online Connection 11-11
11-2 Serial Communications 11-15
11-2-1 Overview of Serial Communications 11-15
11-3 Communications Networks 11-29

11-1 Accessing a PLC from the CX-Programmer

11-1-1 Overview

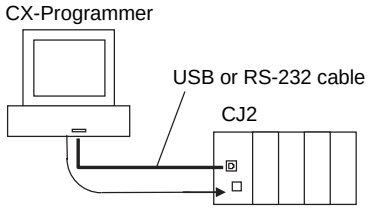
This section provides an overview on how to access a PLC from the CX-Programmer.

Connecting to a PLC

Either of the following two methods can be used to access a PLC from the CX-Programmer.

● Connecting Directly to a PLC

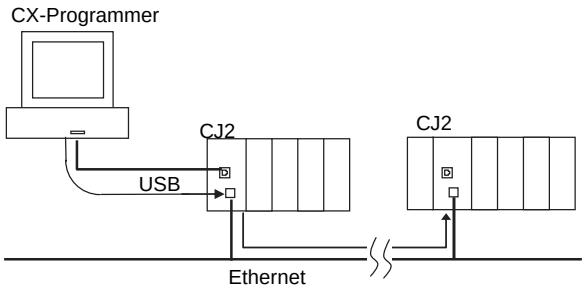
You can connect online to a PLC that is connected directly to the CX-Programmer through a serial port.



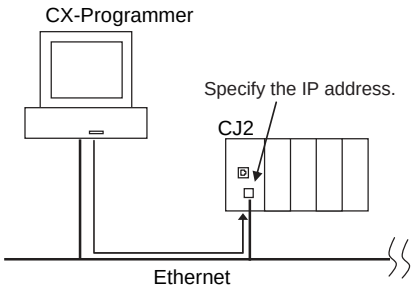
● Accessing a PLC on a Network

You can use either of the following two methods to access a PLC on the network.

Connect online to a PLC on a network through a PLC directly connected to the CX-Programmer through a serial port.



Connect online to a PLC on a network from a personal computer connected directly to the network.



Procedures for Connecting Online to a PLC

Either of the following two procedures can be used to connect online to a PLC from the CX-Programmer.

● Change PLC Dialog Box

Set the method for accessing the target PLC for the CX-Programmer project. This setting is made when creating the project, but it can be changed after the project has been created.

● Automatic Online Connection

An automatic online connection is used to access a PLC for which a connection has been established. It can be used without creating a CX-Programmer project.

Direct Connection

Direct connection is used to connect online to a PLC connected directly to the CX-Programmer through a serial port.

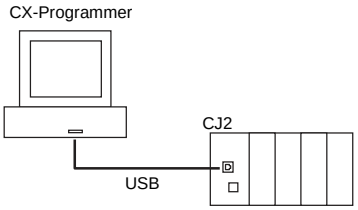
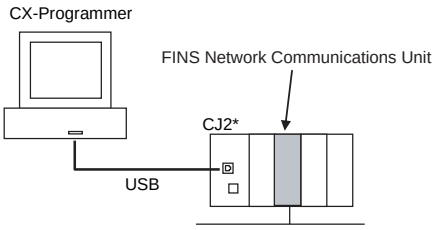
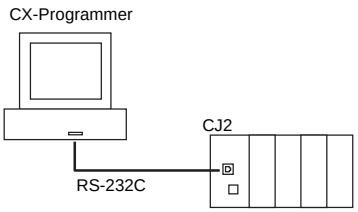
EtherNet/IP Node Connection

An EtherNet/IP node connection is used to access a PLC for which a connection has been established through EtherNet/IP. Even if the IP address of the target PLC is not known, the PLC can be accessed by searching for it.

11-1-2 System Configurations for Accessible PLCs

Direct Serial Connection with PLC

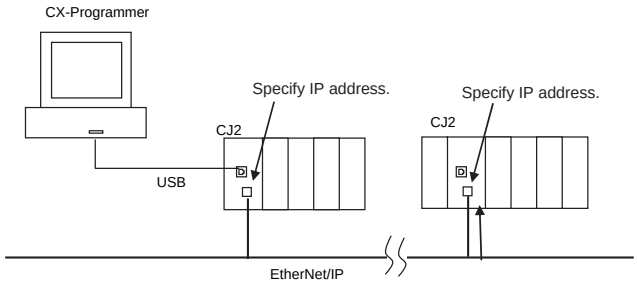
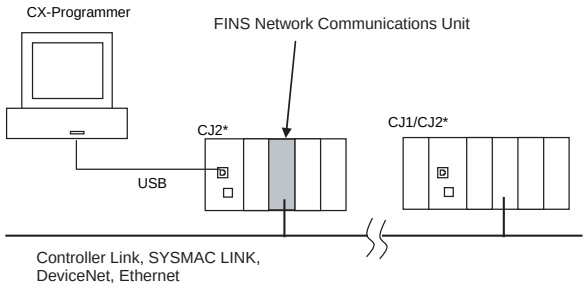
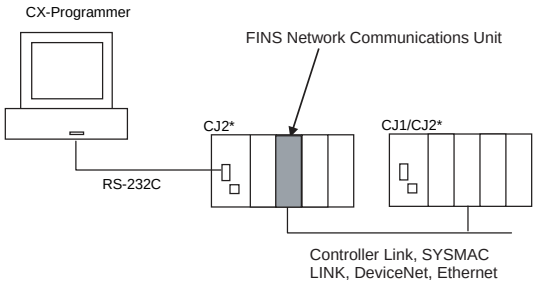
Select a network type as given in the following table.

Connecting Cable	System configuration	Change PLC Dialog Box (network type)	Automatic online connection	
			Direct connection	EtherNet/IP connection
USB cable		USB	Accessible	Not accessible
	Accessing a FINS Network  * The local network table must be set.	Toolbus (USB port)	Accessible	Not accessible
RS-232C cable		Toolbus	Accessible	Not accessible

Accessing a PLC on a Network

● Accessing through a PLC Connected through a Serial Port

Select a network type as given in the following table.

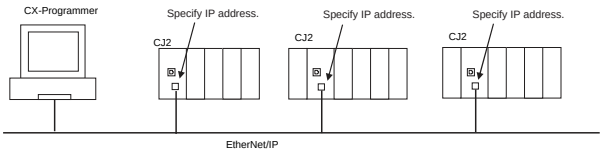
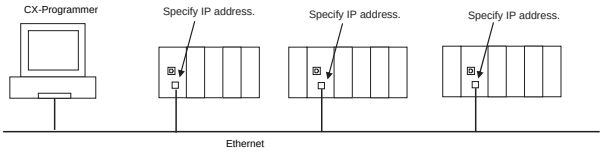
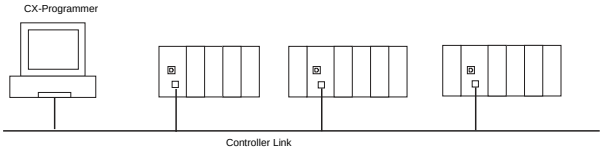
Connecting Cable	System configuration	Change PLC Dialog Box (network type)	Automatic online connection	
			Direct connection	Ether-Net/IP connection
USB cable	Accessing an EtherNet/IP Network via USB (*1)  	USB	Not accessible	Accessible
	Accessing a FINS Network via USB (*2)  * The local network table must be set.	Toolbus (USB port)	Not accessible	Not accessible
RS-232C cable	Accessing a FINS Network via RS-232C (*2)  * The local network table must be set.	Toolbus	Not accessible	Not accessible

*1 If the network type is set in the dialog box for changing the PLC model setting, connection to EtherNet/IP is possible for the local network only. Connection cannot be made across network layers. Connection across network layers is possible if the EtherNet/IP connection is made with an automatic online connection.

*2 It is possible to use connections that cross layers between FINS networks.

Connecting through a Network

Select a network type as given in the following table.

Connecting cable	System configuration	Change PLC Dialog Box (network type)	Automatic online connection	
			Direct connection	EtherNet/IP connection
EtherNet/IP	Accessing an EtherNet/IP Network (*1)  The IP address at the connection target must be set.	EtherNet/IP	Not accessible	Accessible
Ethernet	Accessing Ethernet as a FINS Network (*2)  The FINS network address, node address, and IP address at the connection destination must be set.	Ethernet, Ethernet (FINS/TCP)	Not accessible	Not accessible
Controller Link		Controller Link	Not accessible	Not accessible

*1 If the network type is set in the dialog box for changing the PLC model setting, connection to EtherNet/IP is possible for the local network only. Connection cannot be made across network layers. Connection across network layers is possible if the EtherNet/IP connection is made with an automatic online connection.

*2 It is possible to use connections that cross layers between FINS networks.

Precautions for Correct Use

Connection is not possible when any of the following errors occurs if connection is made with Support Software via the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□.

I/O Bus Errors (Connection is possible for the CJ2M-CPU3□.)

Too many I/O Points Errors

Unit Number Duplication Errors

Rack Number Duplication Errors

I/O Setting Errors

If connection is not possible with the Support Software if any of the errors above occurs, make the connection with the Support Software via the USB port on the CPU Unit.

Precautions for Correct Use

Precautions before Using EtherNet/IP Connections

Better firewall security for Windows XP (SP2 or higher) and Windows Vista has increased the restrictions for data communications on Ethernet ports. When using an EtherNet/IP connection*1 to one of the following PLCs from an EtherNet port on a computer, you must change the settings of the Windows Firewall to enable using CX-Programmer communications.

*1: An EtherNet/IP connection includes the following cases:

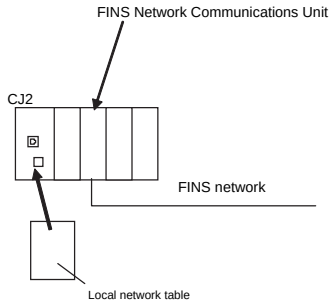
- An online connection with the network type set to EtherNet/IP
- An automatic online connection to a PLCs on an EtherNet/IP network when Auto Online - EtherNet/IP Node Online is selected from the PLC Menu.

Refer to *A-6 EtherNet/IP Connections from Windows XP (SP2 or Higher) or Windows Vista* for operating procedures.

Additional Information

Routing tables must be created and transferred in the following cases.

If one or more FINS Network Communications Units* are connected to a CJ2 CPU Unit, it is necessary to create and transfer the local network table of the routing tables with the CX-Integrator so that it is possible to access both the built-in EtherNet/IP network (on the CJ2H-CPU6□-EIP or CJ2M-CPU3□) and the FINS network from the Support Software.



*A “FINS Network Communications Unit” indicates a Controller Link Unit, SYSMAC LINK Unit, Ethernet Unit, DeviceNet Unit, or FL-net Unit. CompoNet Units are not included. It also indicates using the Serial Gateway for serial communications together with routing tables.

Additional Information

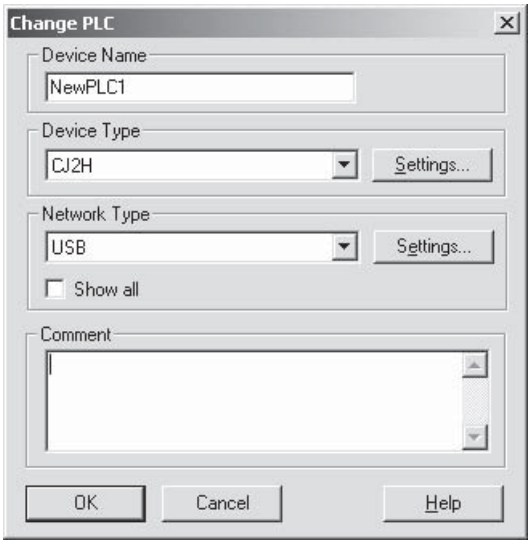
If one or more FINS Network Communications Units are connected to a CJ2H-CPU6□-EIP or CJ2M-CPU3□, create and transfer the local network table. If you attempt access from the Support Software with the network address set to 0 (i.e., default) without creating and transferring the local network table, you will access the node address of the built-in EtherNet/IP port in the following cases.

If the unit number (default: 0) of the built-in EtherNet/IP port is lower than the unit number of the FINS Communications Unit.

11-1-3 Accessing a PLC from the CX-Programmer

Procedures in Change PLC Dialog Box

When creating a new project, use the following procedure in the Change PLC Dialog Box to select the method for connecting to the PLC. This example is for a CJ2H CPU Unit.



1 Select *CJ2H* in the *Device Type* Field.

2 Select the method for connecting to the PLC in the *Network Type* Field. Either of the following methods can be selected as the default.

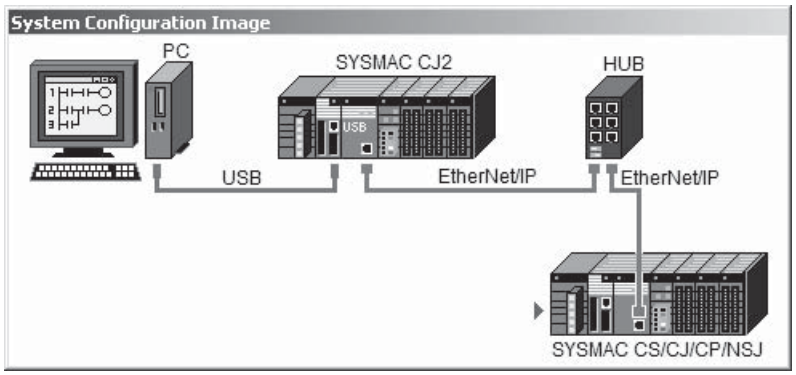
- USB
- EtherNet/IP

If required, other network types can be displayed by clicking **Show All** Button. Select the network type according to the two conditions shown in the following table.

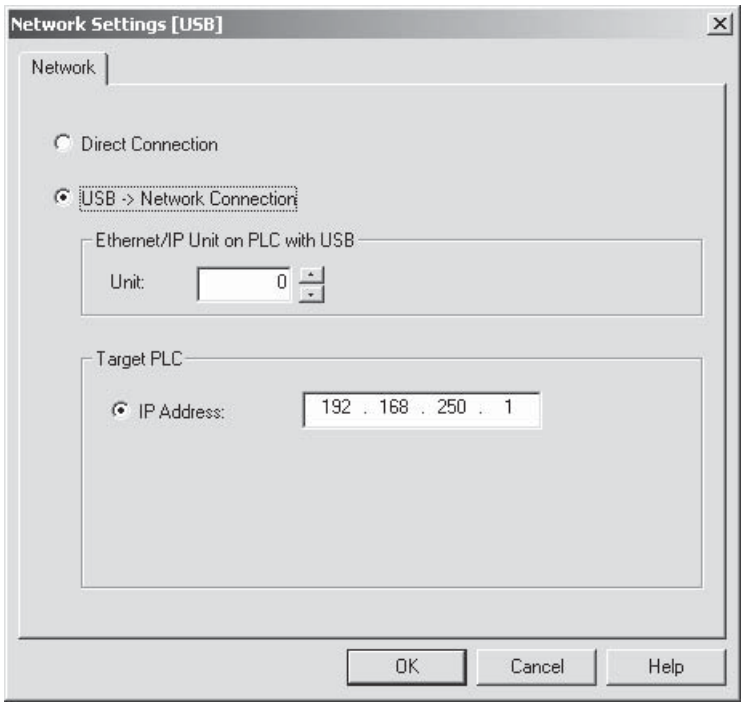
Condition 1: Type of cable connected to the PLC	Condition 2	Network type	
USB	---	USB	Displayed by default.
Ethernet	Accessing an EtherNet/IP network	EtherNet/IP	
USB	Accessing a FINS network via USB	Toolbus (USB port)	Displayed by clicking **Show All** Button.
Ethernet	Accessing an Ethernet network as a FINS network	Ethernet	
		Ethernet (FINS/TCP)	
RS-232C	---	Toolbus	
		SYSMAC WAY	
Controller Link (Connected by Controller Link Board.)	---	Controller Link	
Fins Gateway	---	Fins Gateway	

3 Click the **Settings** Button. A Network Settings Dialog Box will be displayed for the network type that is selected.
In this example, the network types that are displayed by default, i.e., USB and EtherNet/IP, are described. For details on the settings for other network types, refer to the *CX-Programmer Operation Manual* (Cat. No. W446).

● Network Type: USB



Selecting *USB -> Network Connection* in the Network Tab Page



Direct Connection:

Select this option to access a PLC connected directly via USB.

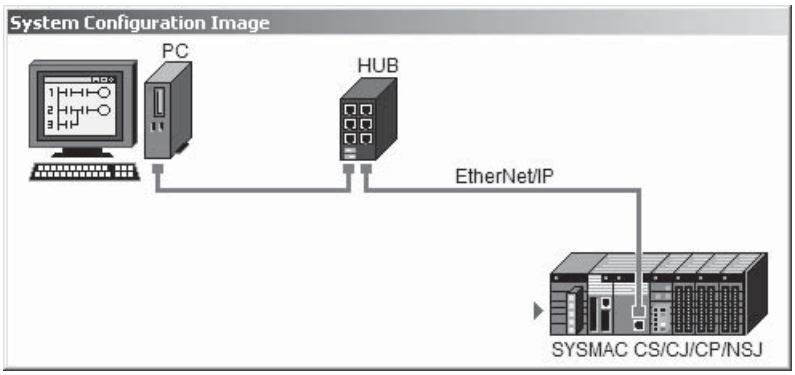
USB -> Network Connection:

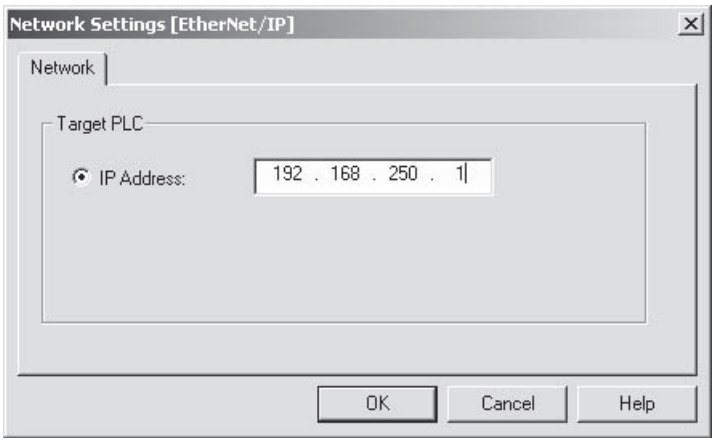
Select this option to access a PLC on an EtherNet/IP network (a single network) through the USB port. Make the following settings:

EtherNet/IP Unit on PLC connected with USB:
When connected to a built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□, set the unit to 0.

IP Address of Target PLC:
Set the IP address of the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ or the EtherNet/IP Unit.
The default IP address for the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ is as follows: 192.168.250.*node_address*.

● Network Type: EtherNet/IP





IP Address of *Target PLC*:

Set the IP address of the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ or the EtherNet/IP Unit.
The default IP address for the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ is as follows: 192.168.250.*node_address*.

● Setting the IP Address of the Personal Computer

When connected to a PLC via EtherNet/IP, the IP address of the personal computer must be set to match the IP address of the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ or the EtherNet/IP Unit. If, for example, the IP address of the built-in EtherNet/IP port on the CJ2H-CPU6□-EIP or CJ2M-CPU3□ is set to the default, set the following IP address for the personal computer.

- **IP Addresses**

Example: 192.168.250.2

- 192.168.250. : Value for same segment as IP address of built-in EtherNet/IP port
- 2 : User-set value (1 to 254)

- **Subnet Mask**

Example: 255.255.255.0.200

- 255.255.255.0 : Class C mask
- 200 : User-set value (0 to 252)

11-1-4 Automatic Online Connection

If the project for the target PLC is not available at the personal computer, it is possible to connect online to upload the programs from the PLC.

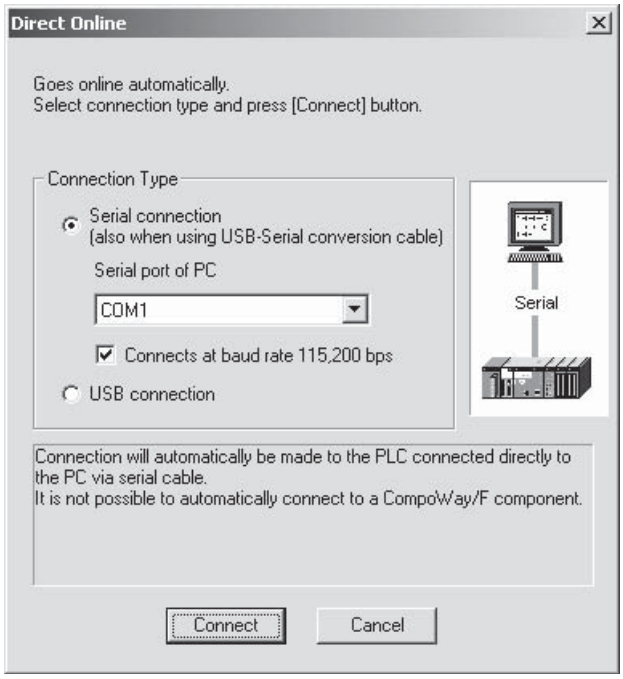
Direct Serial Connection

When an automatic online connection is executed, a search is automatically performed for a usable personal computer serial port. When an applicable serial port is found the CX-Programmer is connected automatically to the PLC connected to the serial port.

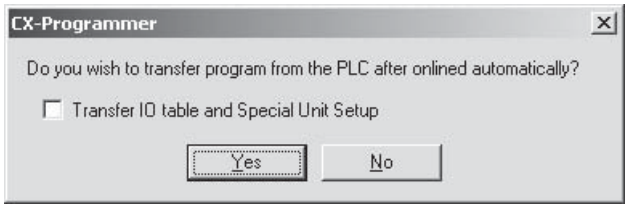
1 Select ***Auto Online*** – ***Direct Online*** from the PLC Menu. The Direct Online Dialog Box will be displayed as shown below.

2 Select either of the following methods for connecting the personal computer to the PLC.

- Serial connection (including converting a USB connection at the personal computer to an RS-232C connecting at the PLC)
- USB connection



When connected to the peripheral USB port of the CJ2 CPU Unit from the USB port of the personal computer, select the USB connection and click the **Connect** Button. The following dialog box will be displayed.



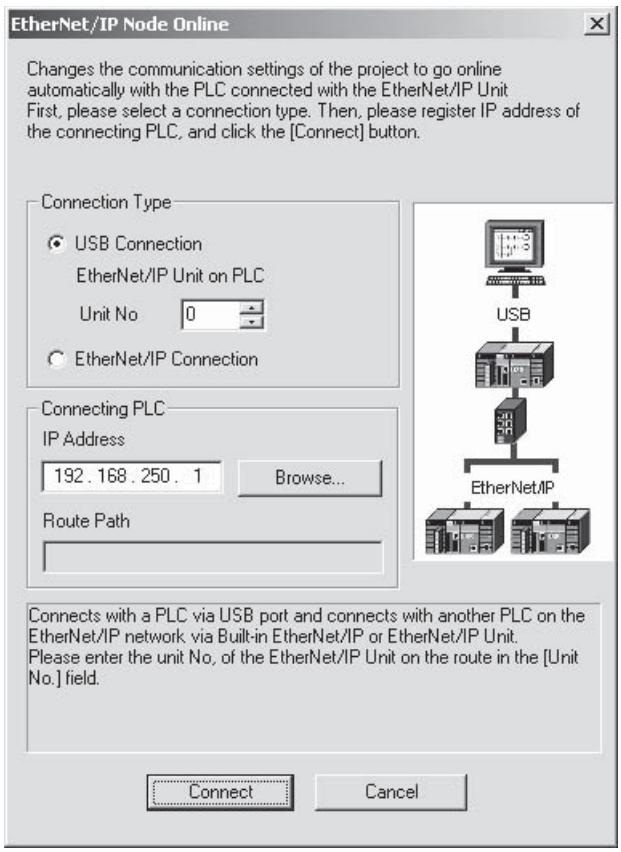
To have the program, PLC Setup, and I/O tables uploaded to the personal computer after automatically connecting online, click the **Yes** Button. The CX-Programmer will be automatically connected to the PLC, and these items will be uploaded from the PLC.

EtherNet/IP Node Connection

It is possible to connect through an EtherNet/IP network to a PLC connected to the EtherNet/IP network. If the IP address of the target PLC is not known, then the PLC can be accessed by searching for it.

1 Select ***Auto online*** – ***EtherNet/IP Node Online*** from the PLC Menu.

2 The EtherNet/IP Node Online Dialog Box will be displayed.



3 For the connection type, select the method for connecting to the target PLC.

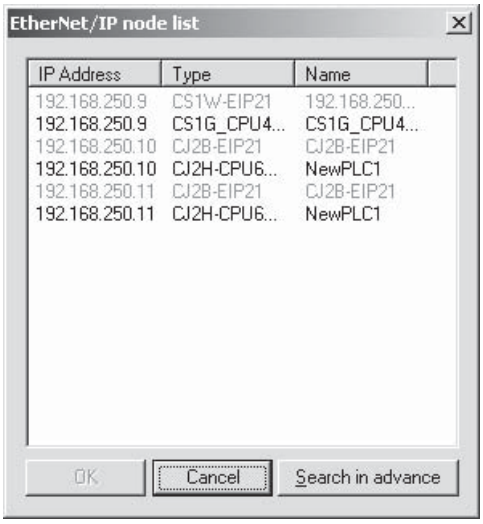
- USB Connection
  Select a USB connection to physically connect directly to the PLC through a USB port and then to connect online to a PLC on an EtherNet/IP network (a single network) via the built-in EtherNet/IP port or an EtherNet/IP Unit.
- EtherNet/IP Connection
  Select an EtherNet/IP connection to physically connect the personal computer to an EtherNet/IP network and to connect online to a PLC on the network (a single network) via EtherNet/IP.

4 For the connecting PLC, select the method for setting the IP address of the target PLC.

- IP Address
  Directly input the IP address of the target PLC.
- Browse
  Search for the PLCs that are connected online to the network, and select the target PLC. Use this method when the IP address of the target PLC is not known.

The connected nodes can be displayed either as a list or by network layer.

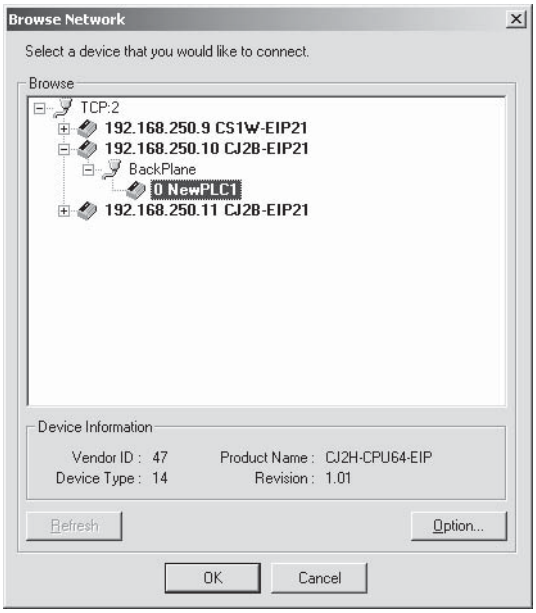
Node List (Displayed for *Browse*)



- The IP address, device type, and name will be displayed. Select the IP address and click the **OK** Button.

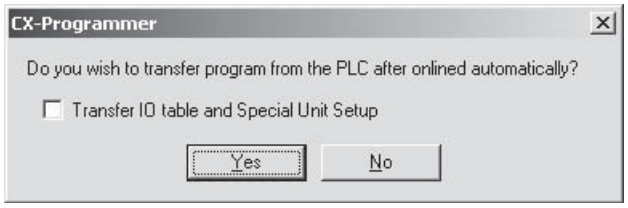
Search in Advance

Click the *Search in advance* Button in the EtherNet/IP node list Dialog Box to display the following dialog box.



- Click a plus (+) icon to display all the levels under that item. Select the PLC to be connected, and click the **OK** Button.

5 Click the **Connect** Button to connect online. The following dialog box will be displayed.



To have the programs, PLC Setup, and I/O tables uploaded to the personal computer after automatically connecting online, click the **Yes** Button. The computer will be automatically connected to the PLC, and these items will be uploaded from the PLC.

11-2 Serial Communications

11-2-1 Overview of Serial Communications

The serial communications port mode (protocol) can be switched in the CPU Unit's PLC Setup. Depending on the protocol selected, the following systems can be configured.

Protocols

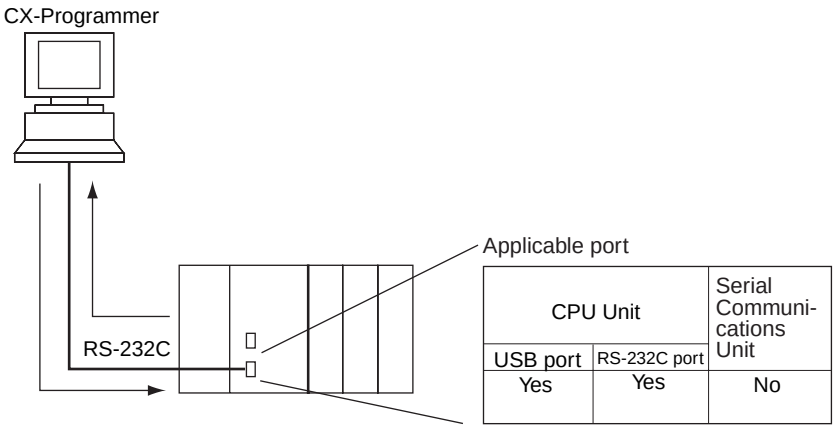
The following protocols support serial communications.

Protocol (Serial Communications Mode)	Main connection	Use	Applicable commands, communications instructions
Peripheral bus (toolbus) (RS-232C port on CPU Unit only. Not supported for Serial Communications Unit.)	CX-Programmer	Communications between Programming Devices and the PLC from the computer.	None
Host Link (SYSMAC WAY) slave	Personal computer OMRON Programmable Terminals	Communications between the Host computer and the PLC. Commands can be sent to a computer from the PLC.	Host Link commands/ FINS commands. Commands can be sent to a computer from the PLC.
No-protocol*1	General-purpose external devices	No-protocol communications with general-purpose devices.	TXD(236) instruction, RXD(235) instruction, TXDU(256) instruction, RXDU(255) instruction DTXDU (262) instruction DRXDU (261) instruction
NT Links (1: N)	OMRON Programmable Terminals	High-speed communications with Programmable Terminals via direct access.	None
Serial Gateway	OMRON Component PLC	Converts FINS commands that are received into CompoWay/F, Modbus, or Host Link protocols, and then transmits the converted command to the serial line.	---
CompoWay/F Master*1	CompoWay/F slave	Converts FINS commands (encapsulated CompoWay/F commands) received at the serial port into CompoWay/F commands.	FINS command 2803 hex received (including sending FINS command using CMND(490))
Modbus Master*1 (Serial Communications Unit only. Not supported for RS-232C port on CPU Unit.)	Modbus slave	Converts FINS commands (encapsulated Modbus commands) received at the serial port into Modbus commands.	FINS command 2804 hex or 2805 hex received (including sending FINS command using CMND(490))
Host Link FINS (SYSWAY) Master*1 (Serial Communications Unit only. Not supported for RS-232C port on CPU Unit.)	Host Link FINS (SYSWAY) slave (PLC)	Converts FINS commands into FINS commands encapsulated in Host Link	Any FINS command received except those sent to serial port (including sending FINS command using CMND(490))
Protocol macro (Serial Communications Unit only. Not supported for RS-232C port on CPU Unit.)	General-purpose external devices	Sending and receiving messages (communications frames) according to the communications specifications of external devices. (SYSMAC-PST is used to create protocols by setting various parameters.)	PMCR(260) instruction
Serial PLC Links (CJ2M CPU Units only)	OMRON PLCs (CJ2M, CJ1M, CP1H, CP1L, and CP1E CPU Units)	Up to 10 words of data per Unit can be shared between up to nine CPU Units (one Polling Unit and up to eight Polled Units).	None

*1 Serial Communications Unit with unit version 1.2 or later only.

Peripheral Bus (Toolbus)

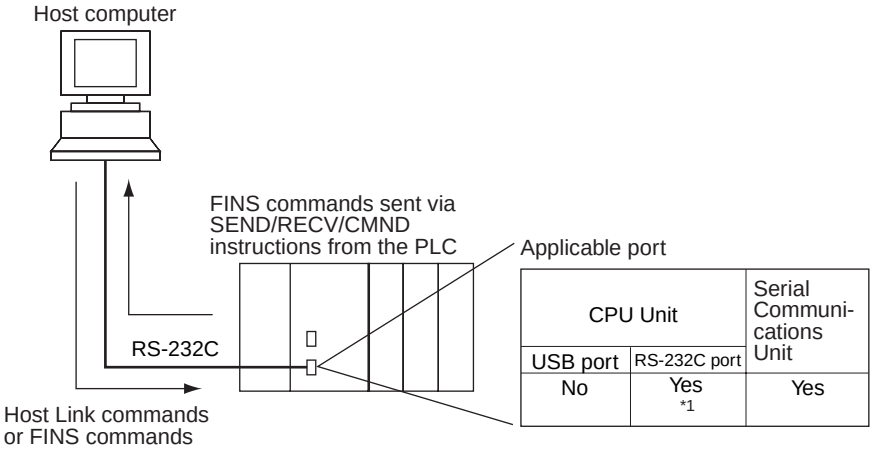
CX-One Support Software, such as the CX-Programmer



CPU Unit		Serial Communications Unit
USB port	RS-232C port	
Yes	Yes	No

Host Link System Slave (SYSMAC WAY Mode 1:N)

The Host Link System allows the I/O memory of the PLC to be read/written, and the operating mode to be changed from a Host computer (personal computer or Programmable Terminal) by executing Host Link commands or FINS commands that are preceded by a header and followed by a terminator. Alternatively, FINS commands (preceded by a header and followed by a terminator) can be sent to a computer connected via the Host Link System by executing Network Communications Instructions (SEND(090)/RECV(098)/CMND(490)) from the PLC.

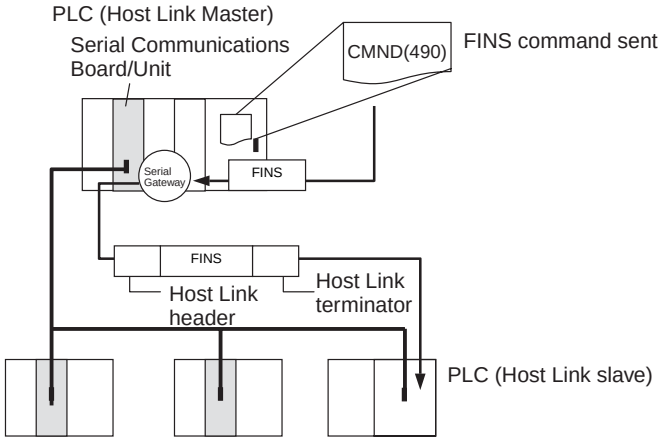


CPU Unit		Serial Communications Unit
USB port	RS-232C port	
No	Yes *1	Yes

*1 Set pin 5 of the DIP switch on the front panel of the CPU Unit to OFF, and set the serial communications mode in the PLC Setup to Host Link.

Additional Information

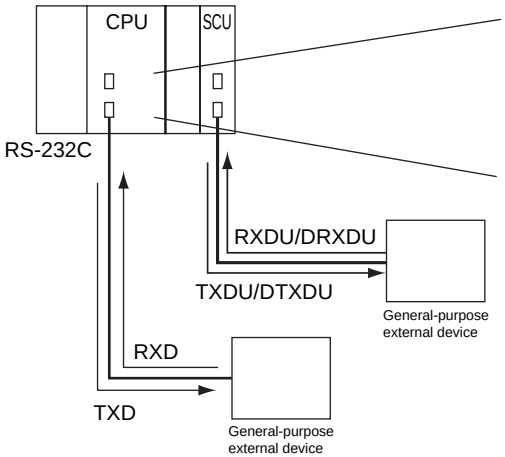
Host Link master functions can be performed by sending the CMND(490) instruction via the Serial Gateway when using Serial Communications Units with unit version 1.2 or later.



No-protocol Communications System

No-protocol communications allow simple data transmissions, such as inputting bar code data and outputting printer data using communications port I/O instructions. The start and completion codes can be set, and RS and CS signal control is also possible with no-protocol communications.

The following figure shows the usage of each communications port I/O instruction, based on the communications port being used and the direction of the data transfer (sending or receiving).



Applicable port

CPU Unit		Serial Communications Unit
USB port	RS-232C port	
No	Yes *1	Yes *2

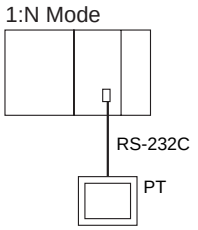
*1 Set pin 5 of the DIP switch on the front panel of the CPU Unit to OFF, and set the serial communications mode in the PLC Setup to no-protocol communications.

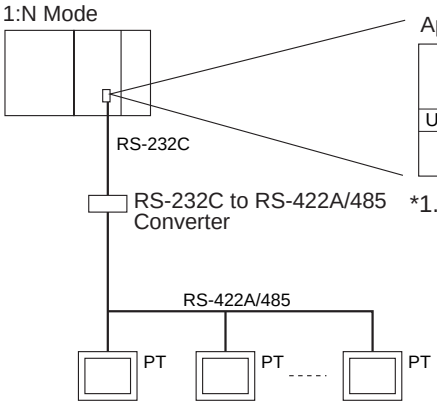
*2 No-protocol communications are supported for Serial Communications Units with unit version 1.2 or later only.

NT Link System (1:N Mode)

If the PLC and Programmable Terminal (PT) are connected together using RS-232C ports, the allocations for the PT's status control area, status notify area, objects such as touch switches, indicators, and memory maps can be allocated in the I/O memory of the PLC. The NT Link System allows the PT to be controlled by the PLC, and the PT can periodically read data from the status control area of the PLC, and perform necessary operations if there are any changes in the area. The PT can communicate with the PLC by writing data to the status notify area of the PLC from the PT. The NT Link system allows the PT status to be controlled and monitored without using PLC ladder programs. The ratio of PLCs to PTs is 1:N, where N is greater than or equal to 1.

Set the PT communications settings for a 1:N NT Link. One or more PTs can be connected to each PLC.





Applicable port

CPU Unit		Serial Communications Unit
USB port	RS-232C	
No	Yes *1	Yes

*1. Turn OFF pin 5 on the DIP switch on the front of the CPU Unit and set the serial communications mode in the PLC Setup for an NT Link.

Precautions for Correct Use

- The PLC can be connected to any PT port that supports 1:N NT Links. It cannot be connected to the RS-232C ports on the NT30 or NT30C, because these ports support only 1:1 NT Links.
- The NT20S, NT600S, NT30, NT30C, NT620S, NT620C, and NT625C cannot be used if the CPU Unit's cycle time is 800 ms or longer (even if only one of these PTs is connected).
- When more than one PT is connected to the same PLC, be sure that each PT is assigned a unique unit number. Malfunctions will occur if the same unit number is set on more than one PT.

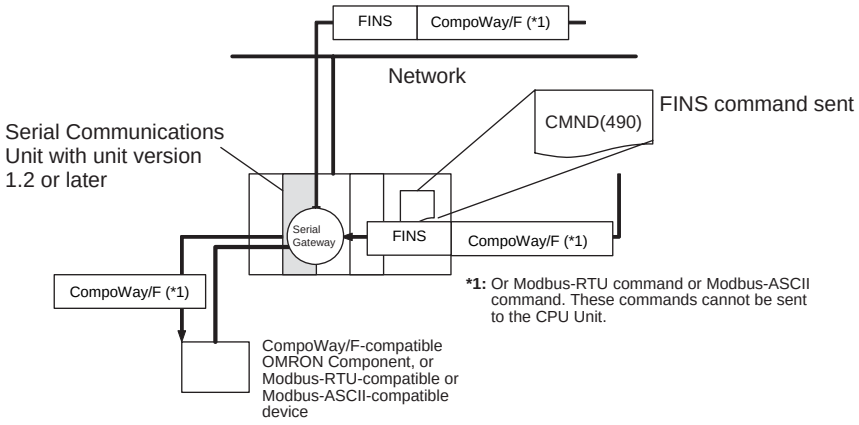
Additional Information

The 1:1 and 1:N NT Link protocols are not compatible with each other, i.e., they are separate serial communications protocols.

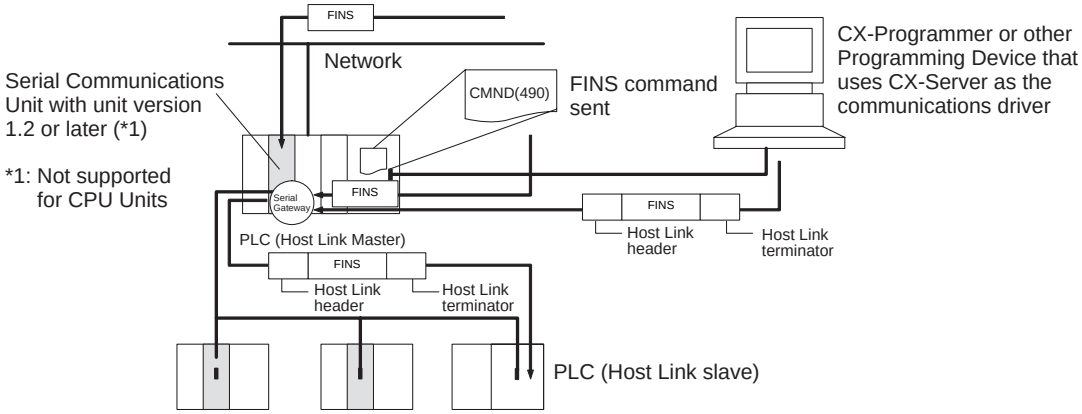
Serial Gateway Mode

The received FINS message is automatically converted into CompoWay/F according to the message. When Serial Communications Units with unit version 1.2 or later are used, the received FINS message is automatically converted into either CompoWay/F, Modbus-RTU, Modbus-ASCII, or Host Link FINS according to the message.

● CompoWay/F, Modbus-RTU, Modbus-ASCII



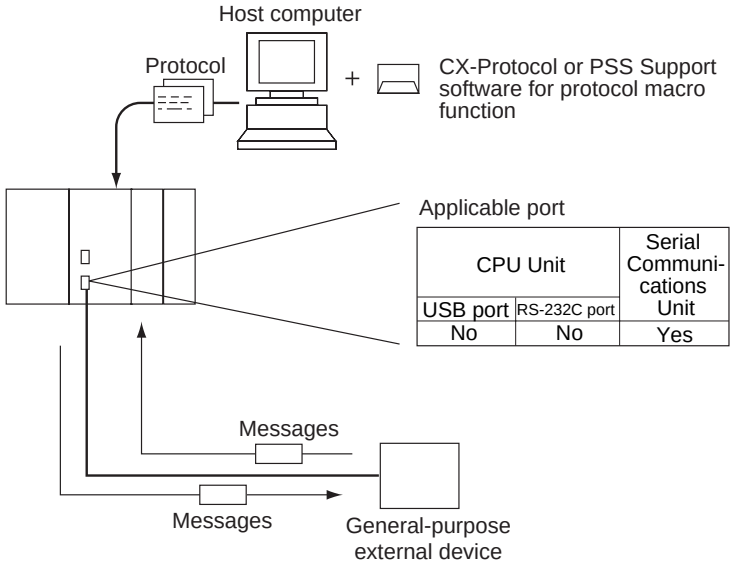
● Host Link FINS



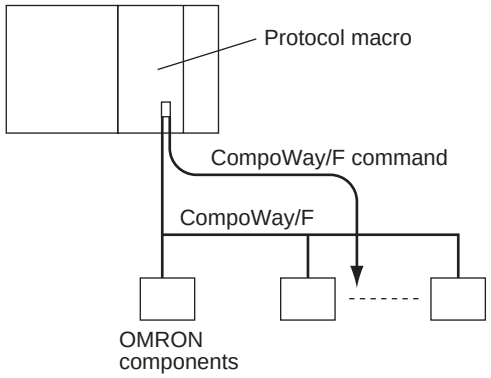
Protocol Macros (Serial Communications Units Only)

The CX-Protocol is used to create data transmission procedures (protocols) for general-purpose external devices according to the communications specifications (half-duplex or full-duplex, asynchronous) of the general-purpose external devices. The protocols that have been created are then recorded in a Serial Communications Unit, enabling data to be sent to and received from the external devices by simply executing the PMCR(260) instruction in the CPU Unit. Protocols for data communications with OMRON devices, such as Temperature Controller, Intelligent Signal Processors, Bar Code Readers, and Modems, are supported as standard protocols. (*1) The user can also change these protocol as required by the application.

*1 The standard protocols are provided with the CX-Protocol and Serial Communications Unit.



Note CompoWay/F (Host Function)
The CJ-series CPU Unit can operate as a host to send CompoWay/F commands to OMRON components connected in the system. CompoWay/F commands are executed by using the CompoWay/F send/receive sequences in the standard protocols of the protocol macro function.



Serial PLC Links (CJ2M CPU Units Only)

Serial PLC Links are supported only by CJ2M CPU Unit. Serial PLC Links enable exchanging data between CJ2M CPU Units or between CJ2M CPU Units and CJ1M/CP1H/CP1L/CP1E CPU Units without special programming. The Serial PLC Link Area (CIO 3100 to CIO 3199) is used. Connect the CPU Units using RS-232C or RS-422A/485. The CJ1W-CIF11 RS-422A Converter is used to convert the RS-232C serial port on the CJ2M-CPU1□ to RS-422A/485.

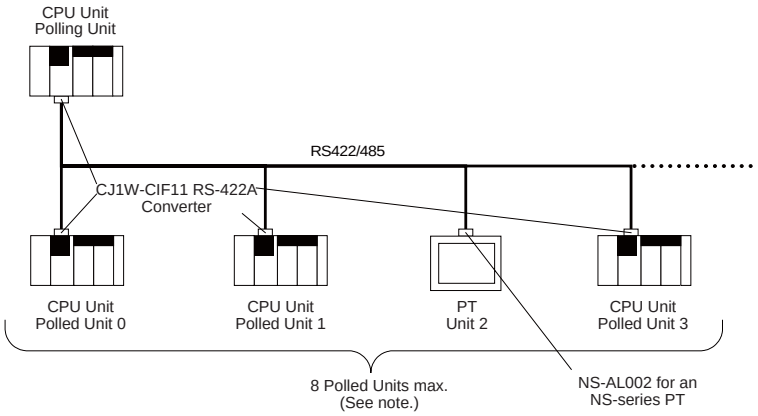
PTs set to the 1:N NT Link protocol can be included in the network. PTs set as Polled Units will use the network to communicate with the CPU Unit set as the Polling Unit via the 1:N NT Link protocol. When connecting to PTs, however, the contents of the words in the Serial PLC Link Area for the PTs will not be stable.

● Operating Specifications

Parameter	Setting
Applicable PLCs	CJ2M, CJ1M, CP1H, CP1L, and CP1E
Baud rate	115,200 or 38,400 bps
Applicable serial ports	Serial port on the CPU Unit (A Serial Option Board is required for the CJ2M-CPU3□.)
Connection method	RS-232C or RS-422A/485
Allocated words	Serial PLC Link Area Words: CIO 3100 to CIO 3199 (Up to 10 words can be allocated for each CPU Unit.) **Note:** CIO 200 to CIO 289 are allocated in CP1E CPU Units for Serial PLC Links.
Maximum number of Units	9 Units max., including 1 Polling Unit and 8 Polled Units. If PT are set to the 1:N NT Link protocol on the same line, the maximum of 9 Unit must include the polled PTs and CPU Units.
Link method (data refresh method)	Complete link method or Polling Unit link method

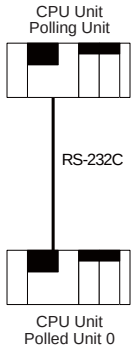
● System Configuration

Connecting CJ2M, CP1L, CP1H, CP1E, or CJ1M CPU Units 1:N (8 Nodes Maximum)



Note: If PTs are set to the Serial PLC Link protocol on the same line, the maximum of 8 Unit must include the polled PTs and CPU Units.

Connecting CJ2M, CP1L, CP1H, CP1E, or CJ1M CPU Units 1:1



Note: If PTs are set to the Serial PLC Link protocol on the same line, the maximum of 8 Unit must include the polled PTs and CPU Units.

Precautions for Correct Use

The CJ1M-CIF11 is not isolated. The maximum transmission distance is therefore 50 m. If the distance exceeds 50 m, use the isolated NT-AL001 and not the CJ1W-CIF11. If you use only the NT-AL001, the maximum total transmission distance will be 500 m.

● Procedure

Serial PLC Links operate based on the following settings in the PLC Setup for the Polling Unit and the Polled Units.

Polling Unit Settings

1 Set the serial communications mode of the RS-232C port to Serial PLC Links, Polling Unit.

2 Set the link method to *All or Polling Unit*.

3 Set the number of words to link. (1 to 10)

4 Set the highest unit number to use for Serial PLC Links. (0 to 7)

Polled Unit Settings

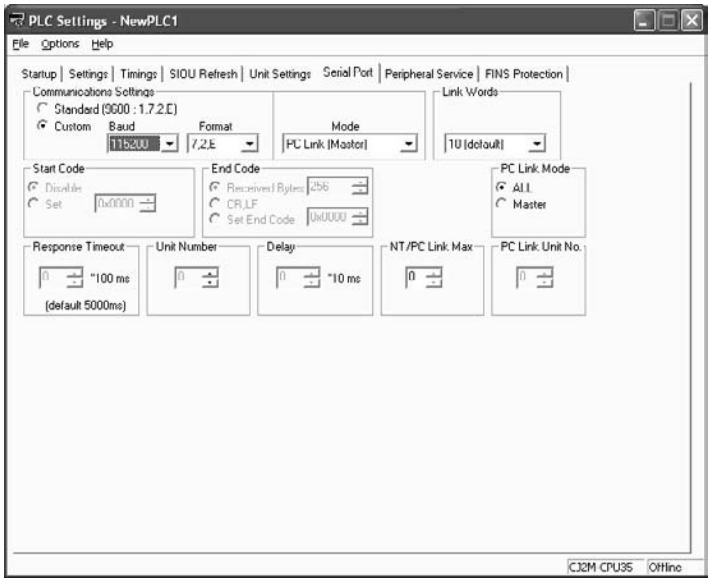
1 Set the serial communications mode of the RS-232C port to Serial PLC Links, Polled Unit.

2 Set the unit numbers of the Polled Units in the Serial PLC Links.

● PLC Setup

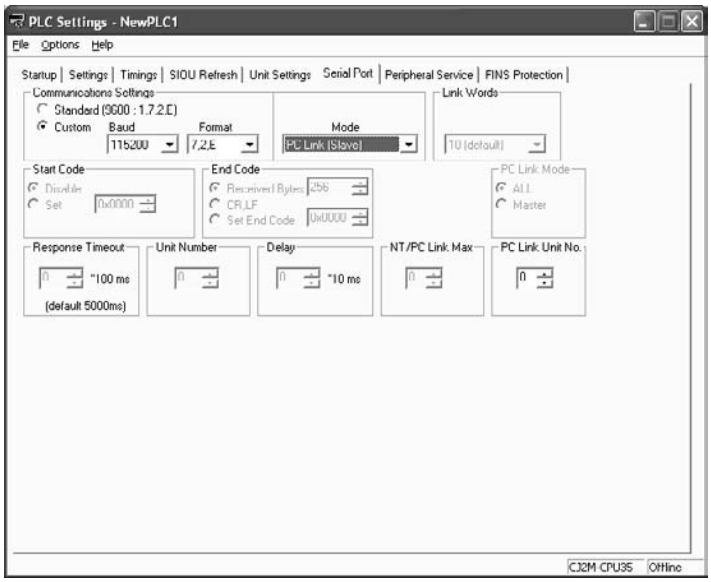
Settings at the Polling Unit

Serial Port Tab Page



Parameter	Setting
Communications Settings	Set the communications settings to match those of the connected PLC. If the connected PLC is set to 115,200 bps, select the Custom Option and set the baud rate to 115,200. It does not matter what is selected for the Parameter setting.
Mode	Select PC Link Polling Unit.
Number of Link Words	This parameter is set only in the Polling Unit. 10 words (default)
PC Link Method	Select *All or Polling Unit.*
Highest Unit Number for NT/Serial PLC Link (No. NT/PC Link Max.)	Set the highest unit number of the connected Polled Units.

Settings at the Polled Units



Parameter	Setting
Communications Settings	Set the communications settings to match those of the connected PLC. If the connected PLC is set to 115,200 bps, select the Custom Option and set the baud rate to 115,200. It does not matter what is selected for the Parameter setting.
Mode	Select PC Link Polled Unit.
Serial PLC Link Unit Number	Set the unit number (0 to 7).

● Link Method (Data Refresh Method)

The following two methods can be used to refresh data.

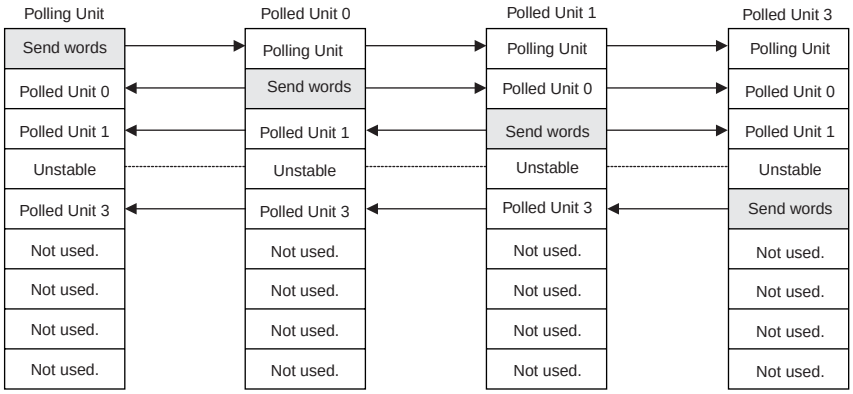
- Complete Link Method
- Polling Unit Link Method

Complete Link Method

The data from all nodes in the Serial PLC Links are updated in both the Polling Unit and the Polled Units. The only exceptions are the data for Polled Units that are not present in the network and the data for any PTs that are connected. These data areas are unstable in all nodes.

Example: Complete Link Method, Highest Unit Number: 3

In the following diagram, Polled Unit 2 is a Unit not present in the network or a PT, so the words allocated for Polled Unit 2 are unstable in all nodes.

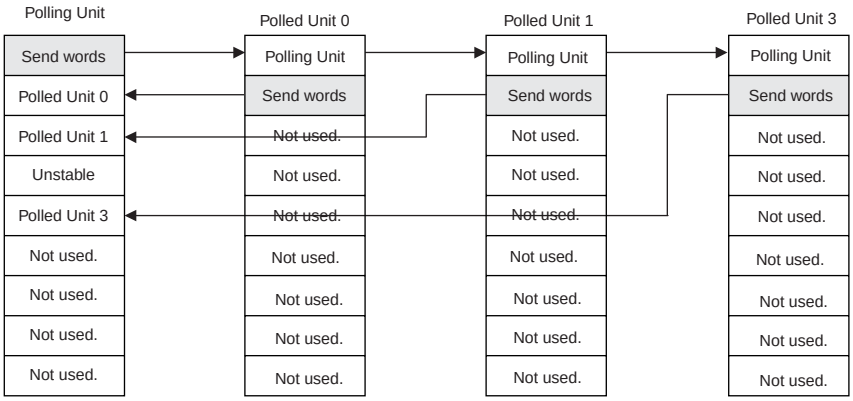


Polling Unit Link Method

The data for all the Polled Units in the Serial PLC Links are updated in the Polling Unit only, and each Polled Unit updates the data of the Polling Unit only. The advantage of the Polling Unit link method is that the addresses allocated for the local Polled Unit data are the same in each Polled Unit, allowing data to be accessed using common ladder programming. The data for polled units that are not present in the network and the data for any PTs that are connected are unstable in all nodes.

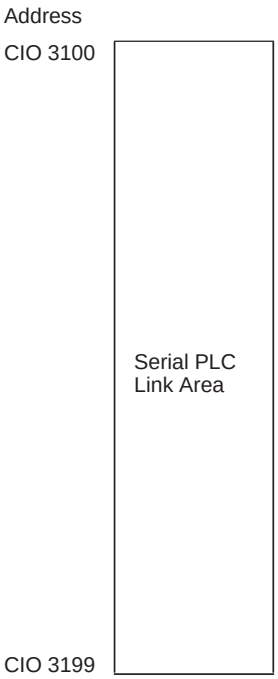
Example: Polling Unit Link Method, Highest Unit Number: 3

In the following diagram, polled unit 2 is a Unit not present in the network or a PT, so this data is unstable in the Polling Unit.



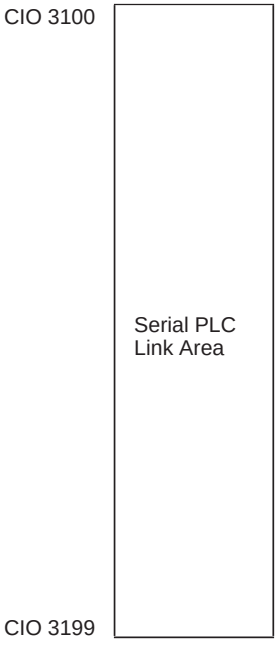
● Allocated Words

Complete Link Method



Number of link words	1 word	2 words	3 words	to	10 words
Polling Unit	CIO 3100	CIO 3100 to CIO 3101	CIO 3100 to CIO 3102		CIO 3100 to CIO 3109
Polled Unit 0	CIO 3101	CIO 3102 to CIO 3103	CIO 3103 to CIO 3105		CIO 3110 to CIO 3119
Polled Unit 1	CIO 3102	CIO 3104 to CIO 3105	CIO 3106 to CIO 3108		CIO 3120 to CIO 3129
Polled Unit 2	CIO 3103	CIO 3106 to CIO 3107	CIO 3109 to CIO 3111		CIO 3130 to CIO 3139
Polled Unit 3	CIO 3104	CIO 3108 to CIO 3109	CIO 3112 to CIO 3114		CIO 3140 to CIO 3149
Polled Unit 4	CIO 3105	CIO 3110 to CIO 3111	CIO 3115 to CIO 3117		CIO 3150 to CIO 3159
Polled Unit 5	CIO 3106	CIO 3112 to CIO 3113	CIO 3118 to CIO 3120		CIO 3160 to CIO 3169
Polled Unit 6	CIO 3107	CIO 3114 to CIO 3115	CIO 3121 to CIO 3123		CIO 3170 to CIO 3179
Polled Unit 7	CIO 3108	CIO 3116 to CIO 3117	CIO 3124 to CIO 3126		CIO 3180 to CIO 3189
Not used.	CIO 3109 to CIO 3199	CIO 3118 to CIO 3199	CIO 3127 to CIO 3129		CIO 3190 to CIO 3199

Polling Unit Link Method



Number of link words	1 word	2 words	3 words	to	10 words
Polling Unit	CIO 3100	CIO 3100 to CIO 3101	CIO 3100 to CIO 3102		CIO 3100 to CIO 3109
Polled Unit 0	CIO 3101	CIO 3102 to CIO 3103	CIO 3103 to CIO 3105		CIO 3110 to CIO 3119
Polled Unit 1	CIO 3102	CIO 3104 to CIO 3105	CIO 3106 to CIO 3108		CIO 3120 to CIO 3129
Polled Unit 2	CIO 3103	CIO 3106 to CIO 3107	CIO 3109 to CIO 3111		CIO 3130 to CIO 3139
Polled Unit 3	CIO 3104	CIO 3108 to CIO 3109	CIO 3112 to CIO 3114		CIO 3140 to CIO 3149
Polled Unit 4	CIO 3105	CIO 3110 to CIO 3111	CIO 3115 to CIO 3117		CIO 3150 to CIO 3159
Polled Unit 5	CIO 3106	CIO 3112 to CIO 3113	CIO 3118 to CIO 3120		CIO 3160 to CIO 3169
Polled Unit 6	CIO 3107	CIO 3114 to CIO 3115	CIO 3121 to CIO 3123		CIO 3170 to CIO 3179
Polled Unit 7	CIO 3108	CIO 3116 to CIO 3117	CIO 3124 to CIO 3126		CIO 3180 to CIO 3189
Not used.	CIO 3109 to CIO 3199	CIO 3118 to CIO 3199	CIO 3127 to CIO 3129		CIO 3190 to CIO 3199

Related Auxiliary Area Flags and Words

Name	Address	Function	Read/Write	Refresh timing
Serial Port PT Communications Flags*	A393.00 to A393.07	The corresponding bit will be ON when the serial port is communicating in NT Link Mode or in Serial PLC Link Mode. Bits 0 to 7 correspond to Units 0 to 7. ON: Communicating OFF: Not communicating	Read	• Cleared when power is turned ON. • Updated if the serial port is in NT Link or Serial PLC Link mode. • Bits 0 to 7 correspond to Units 0 to 7.
Serial Port Restart Bit	A526.00	Turn ON to restart the serial port.	Read/Write	• Cleared when power is turned ON. • Turn ON this bit to restart the serial port in any mode except for Toolbus mode. **Note:** This bit is turned OFF automatically when the restart processing is completed.
Serial Port Error Flags	A528.00 to A528.07	These flags indicate what kind of error has occurred at serial port. Bit 0: Not used. Bit 1: Not used. Bit 2: Parity error Bit 3: Framing error Bit 4: Overrun error Bit 5: Timeout error Bit 6: Not used. Bit 7: Not used.	Read/Write	• Cleared when power is turned ON. • These flags indicate what kind of error has occurred at the serial port. • They are automatically turned OFF by the system when the serial port is restarted. • Only the following bits are valid in Serial PLC Link Mode. Polling Unit: Bit 5: ON for timeout error. Polled Units: Bit 3: ON for framing error. Bit 4: ON for overrun error. Bit 5: ON for timeout error.
Serial Port Settings Changing Flag	A619.02	Turns ON when the communications parameters are being changed for the serial port. ON: Changing OFF: Not changing	Read/Write	• Cleared when power is turned ON. • Turns ON when the communications parameters are being changed for the serial port. • Turns ON when the CHANGE SERIAL PORT SETUP (STUP(237)) instruction is executed. • Turns OFF when changing the parameters has been completed normally.

* In the same way as for the existing 1:N NT Link, the status (communicating/not communicating) of the Polled Unit in Serial PLC Links can be checked from the Polling Unit (CPU Unit) by reading the Serial Port PT Communications Flags (A393.00 to A393.07 for unit numbers 0 to 7).

Unit/Protocol Compatibility

Unit	Model	Port	Protocol (serial communications mode)						
			Peripheral bus (toolbus)	Host Link	No-protocol communications	Protocol macro	NT Link (1:N Mode)	Serial Gateway*2	Serial PLC Link*1
CPU Units	CJ2H-CPU68(-EIP) CJ2H-CPU67(-EIP) CJ2H-CPU66(-EIP) CJ2H-CPU65(-EIP) CJ2H-CPU64(-EIP)	RS-232C	Yes	Yes	Yes	---	Yes	Yes	No
	CJ2M-CPU35 CJ2M-CPU34 CJ2M-CPU33 CJ2M-CPU32 CJ2M-CPU31	RS-232C or RS-422A/485*1	Yes	Yes	Yes	---	Yes	Yes	Yes
	CJ2M-CPU15 CJ2M-CPU14 CJ2M-CPU13 CJ2M-CPU12 CJ2M-CPU11	RS-232C	Yes	Yes	Yes	---	Yes	Yes	Yes
Serial Communications Unit	CJ1W-SCU41-V1 CJ1W-SCU31-V1 CJ1W-SCU21-V1 CJ1W-SCU22 CJ1W-SCU32 CJ1W-SCU42	RS422/485	---	Yes	Yes*2	Yes	Yes	Yes	No
		RS-232C	---	Yes	Yes*2	Yes	Yes	Yes	No

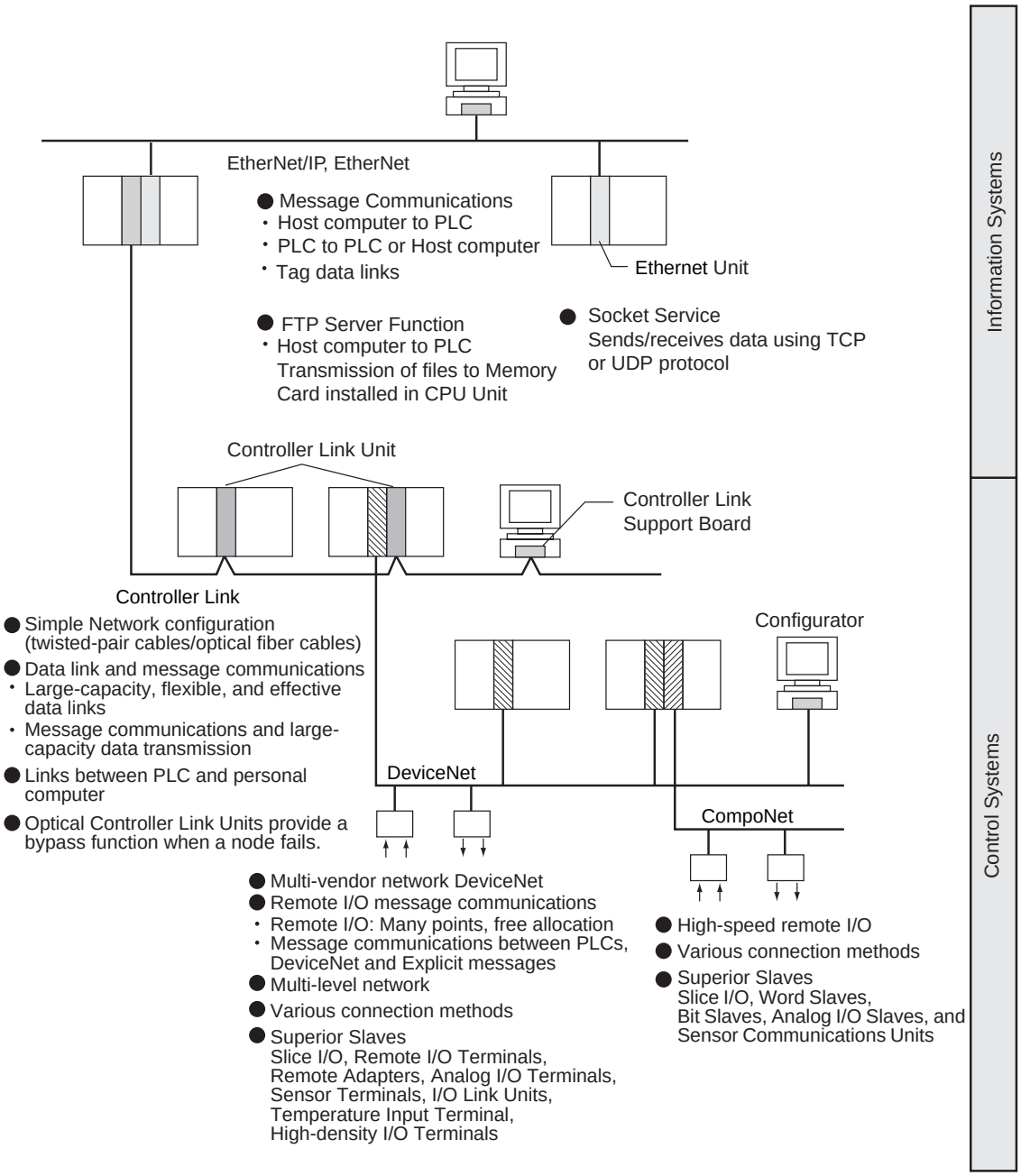
*1 A Serial Option Board can be mounted to enable RS-232C or RS-422A/485 communications.

*2 Supported for Serial Communications Units with unit version 1.2 or later only. For CPU Units, however, only automatic conversion to CompoWay/F is possible for the Serial Gateway protocol.

11-3 Communications Networks

Communications Network Configuration

The following networks can be configured when using CJ-series PLCs.

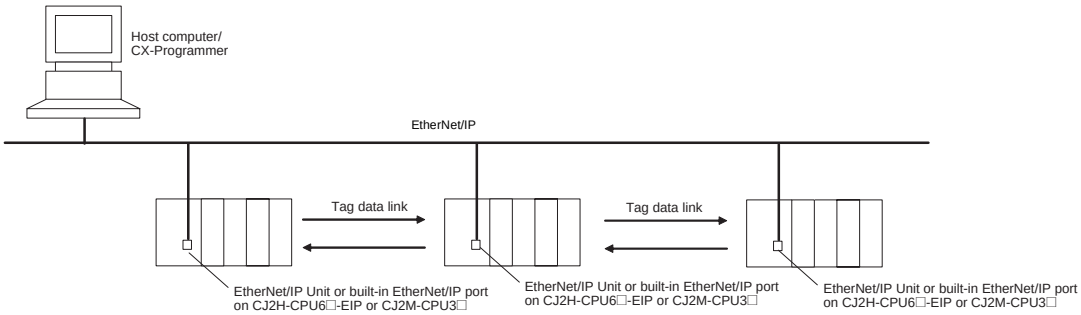


● EtherNet/IP and Ethernet

Tag data links and FINS network communications are enabled over Ethernet cable. In addition, by executing FTP commands for the PLC from the host computer connected to the Ethernet, the contents of the files on the Memory Card installed in the CPU Unit can be read or written. Data can be sent and received using UDP and TCP protocols. These functions enable a greater compatibility with host information networks.

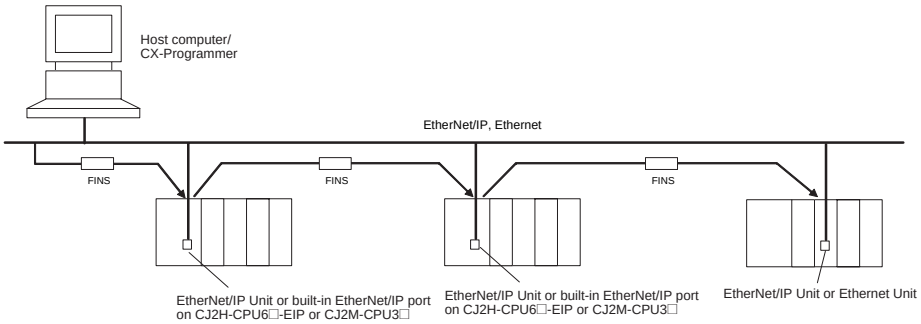
Tag Data Links

Data links can be executed between two PLCs for high-speed, large-capacity data transfers with user-set refresh periods for each area.



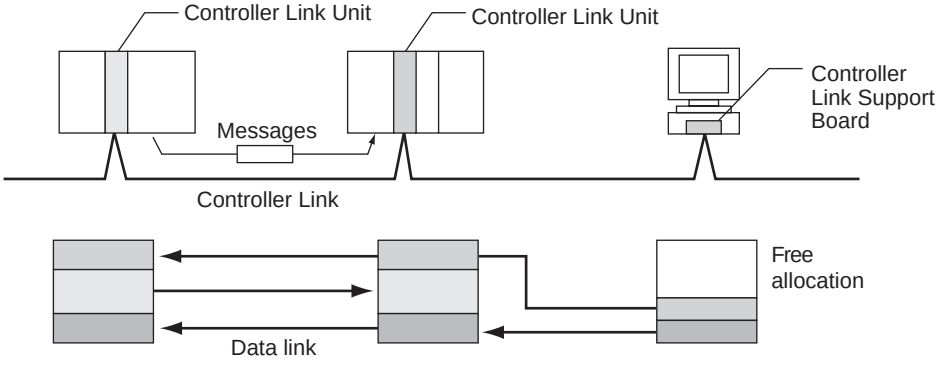
FINS Network Communications

Connecting to the built-in EtherNet/IP port or to a CPU Unit connected to an EtherNet/IP Unit or Ethernet Unit enables FINS message communications with other PLCs or between the PLC and a host computer on the Ethernet network.



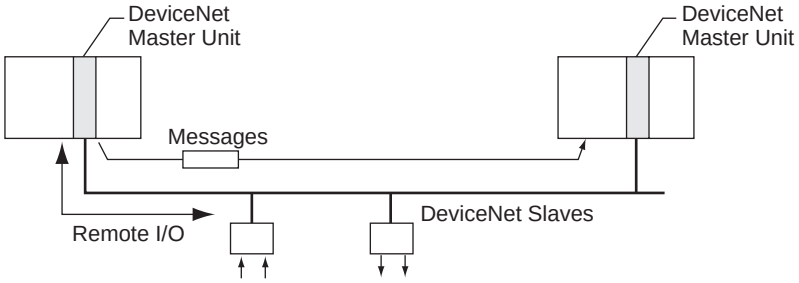
● Controller Link

The Controller Link Network is a special network for OMRON PLC FA. Connecting a Controller Link Unit to the network enables data links between PLCs, so that data can be shared without programming, and FINS message communications between PLCs, which enable separate control and data transfer when required. The Controller Link Network connections use either twisted-pair cables or optical fiber cables. Data links and message communications are also possible between the PLC and personal computer. Data links enable large-capacity and free allocations. FINS message communications also allow large-capacity data transfer.



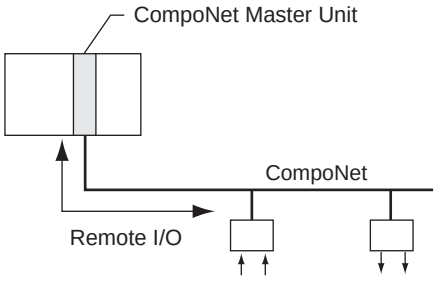
DeviceNet

DeviceNet is a multi-vendor network consisting of multi-bit control and information systems and conforms to the Open Field DeviceNet specification. Connecting a DeviceNet Master Unit to the network enables remote I/O communications between the PLC and the Slaves on the network. Remote I/O communications enable large-capacity I/O and user-set allocations. Analog I/O Terminals are used for the Slaves. Message communications are possible between PLCs and between the PLC and DeviceNet devices manufactured by other companies.



CompoNet

CompoNet is a high-speed, multi-point, multi-node multi-vendor network used mainly for I/O control. Connecting a CompoNet Master Unit in the PLC enables remote I/O communications between the PLC and CompoNet Slaves. High-speed communications are performed with 1,024 points in a cycle time of 1 ms max. Up to 2,560 points and 384 nodes can be connected.



Communications Network Overview

System	Network	Function	Communications	Communications Device
Information networks	EtherNet/IP	Between Host computer and PLC.	FINS message communications	CJ2H-CPU6□-EIP or CJ2M-CPU3□ Built-in EtherNet/IP port EtherNet/IP Unit
		Between PLCs.		
		Between Host computer and Memory Card installed in CPU Unit.	FTP server	
	Controller Link	Between PLC and personal computer directly connected to the Network.	FINS message communications	Controller Link Support Board or Controller Link Unit
			Data link (offset, simple settings)	
	RS-232C → Controller Link	Between Host Link computer and PLC on the Network.	Host Link commands and gateway.	RS-232C cables and Controller Link Unit
Control networks	EtherNet/IP	Between PLCs.	Tag data links	CJ2H-CPU6□-EIP Built-in EtherNet/IP port EtherNet/IP Unit
	Controller Link	Between PLCs.	FINS message communications	Controller Link Unit
			Data link (offset, simple settings)	
	DeviceNet		FINS message communications in an open network.	DeviceNet Master Unit and Configurator
	DeviceNet	PLC and Network devices (Slaves).	Large-capacity remote I/O (fixed or free allocation) in an open network	DeviceNet Master Unit and Configurator
	CompoNet		High-speed, multi-point, multi-node remote I/O in an open network	CompoNet Master Unit

12

CPU Unit Cycle Time

This section describes the cycle time used for processing in the CPU Unit.

12-1 Monitoring the Cycle Time 12-2
12-1-1 Monitoring the Cycle Time 12-2
12-2 Computing the Cycle Time 12-4
12-2-1 CPU Unit Operation Flowchart 12-4
12-2-2 Cycle Time Overview 12-5
12-2-3 I/O Unit Refresh Times for Individual Units 12-7
12-2-4 Cycle Time Calculation Example 12-11
12-2-5 Online Editing Cycle Time Extension 12-13
12-2-6 I/O Response Time 12-13
12-2-7 Response Time for Built-in Input Interrupts 12-14
12-2-8 Response Performance of Serial PLC Links 12-15

12-1 Monitoring the Cycle Time

12-1-1 Monitoring the Cycle Time

The average, maximum, and minimum cycle times can be monitored when the CX-Programmer is connected online to a CPU Unit.

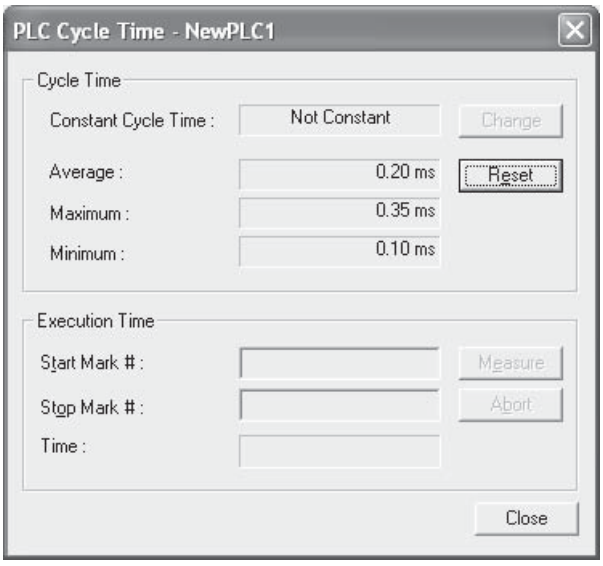
Monitoring the Average Value

While connected online to the PLC, the average cycle time is displayed in the status bar when the CPU Unit is in any mode other than PROGRAM mode.



Monitoring Maximum and Minimum Values

Select ***Edit*** – ***Cycle Time*** from the PLC Menu. The following PLC Cycle Time Dialog Box will be displayed.



The average (mean), maximum, and minimum cycle times are displayed in order from the top. Click the **Reset** Button to recalculate and display the cycle time values.

Additional Information

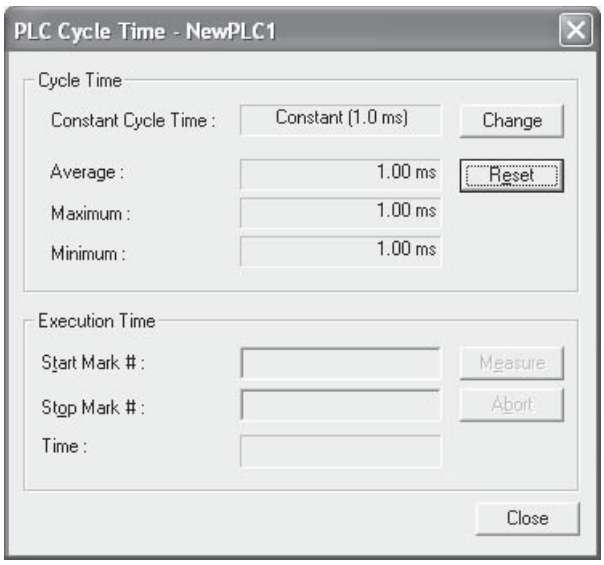
The cycle time average value (= present value) and maximum value are stored in the following Auxiliary Area words.

- Present cycle time (= present value) (Unit: 0.1 ms)
  A264 (lower bytes) and A265 (upper bytes)
- Present cycle time (= present value) (Unit: 0.01 ms)
  A266 (lower bytes) and A267 (upper bytes)
- Maximum cycle time (Unit: 0.01 ms)
  A262 (lower bytes) and A263 (upper bytes)

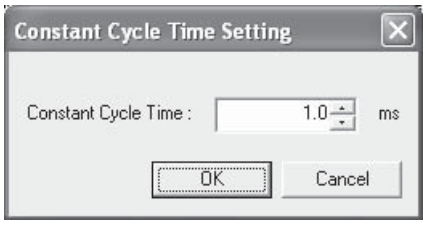
Changing the Minimum Cycle Time

When the minimum cycle time is set, the minimum cycle time can be changed while the CPU Unit is in MONITOR mode.

1 Select ***PLC Info - Cycle Time*** from the PLC Menu. The PLC Cycle Time Dialog Box will be displayed as shown below.



2 Click the **Change** Button. The Constant Cycle Time Setting Dialog Box will be displayed.

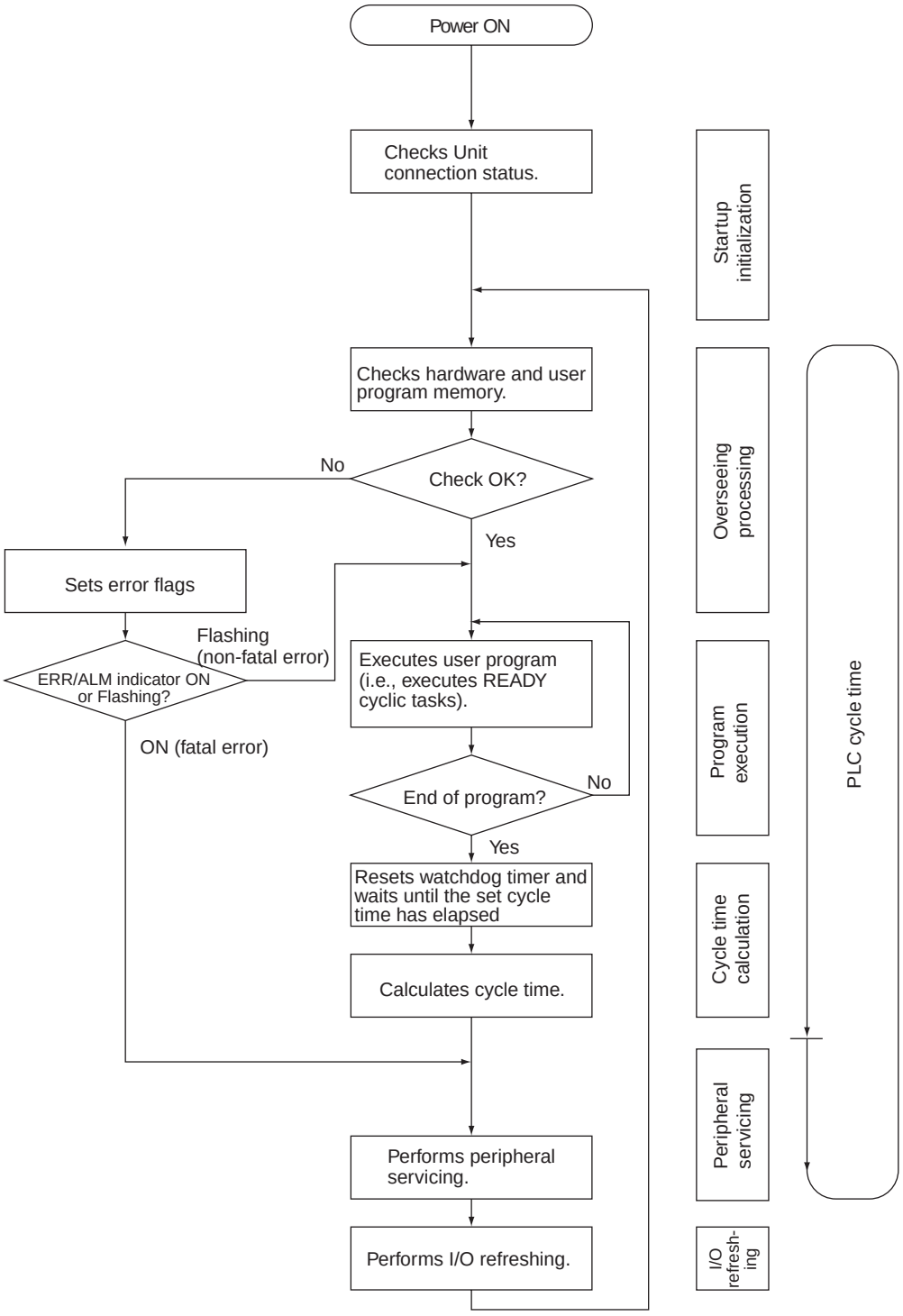


3 Change the time in the *Constant Cycle Time* Field and then click the **OK** Button. The minimum cycle time will be changed.

12-2 Computing the Cycle Time

12-2-1 CPU Unit Operation Flowchart

The CJ-series CPU Units process data in repeating cycles from the overseeing processing up to I/O refreshing as shown in the following diagrams.



12-2-2 Cycle Time Overview

The cycle time depends on the following conditions.

- Type and number of instructions in the user program (in all cyclic tasks that are executed during a cycle, and within interrupt tasks for which the execution conditions have been satisfied).
- Type and number of Basic I/O Units
- Type and number of Special I/O Units and CPU Bus Units.
- Specific servicing for the following Unit/Boards
  - EtherNet/IP tag data link refreshing and the number of tag data link words exchanged
  - Data link refreshing and the number of data link words for Controller Link
  - Remote I/O for DeviceNet Units and the number of remote I/O words
  - Use of protocol macros and the largest communications message
  - Socket services for specific control bits for Ethernet Units and the number of send/receive words
- Fixed cycle time setting in the PLC Setup
- File access in file memory, and the amount of data transferred to/from file memory
- Event servicing for Special I/O Units, CPU Bus Units, and communications ports
- Usage of peripheral USB port and serial ports
- Fixed peripheral service execution time in the PLC Setup

Note 1 The cycle time is not affected by the number of tasks that are used in the user program. The tasks that affect the cycle time are those cyclic tasks that are READY status in the cycle.

2 When the mode is switched from MONITOR mode to RUN mode, the cycle time will be extended by 10 ms (this will not, however, take the cycle time over its limit).

3 The performances of the EtherNet/IP port built into the CJ2H-CPU6□-EIP and the EtherNet/IP port built into the CJ2M-CPU3□ are very different in terms of data processing for EtherNet/IP tag data links. Refer to the *EtherNet/IP Unit Operation Manual* (Cat. No. W465) for details.

The cycle time is the total time required for the PLC to perform the 5 operations shown in the following tables.

Cycle time = (1) + (2) + (3) + (4) + (5)

1. Overseeing

Details	Processing time and fluctuation cause
Checks the I/O bus and user program memory, checks for battery errors and refreshes the clock.	CJ2H-CPU6□-EIP: 0.2 ms CJ2H-CPU6□: 0.1 ms CJ2M-CPU1□: 0.16 ms CJ2M-CPU3□: 0.27 ms

2. Program Execution

Details	Processing time and fluctuation cause
Executes the user program, and calculates the total time taken for the instructions to execute the program.	Total instruction execution time

3. Cycle Time Calculation

Details	Processing time and fluctuation cause
Waits for the specified cycle time to elapse when a minimum (fixed) cycle time has been set in the PLC Setup. Calculates the cycle time.	When the cycle time is not fixed, the time for step 3 is approximately 0. When the cycle time is fixed, the time for step 3 is the preset fixed cycle time minus the actual cycle time ((1) + (2) + (4) + (5)).

4. I/O Refreshing

Details			Processing time and fluctuation cause
Basic I/O Units	Basic I/O Units are refreshed. Outputs from the CPU Unit to the Output Units are refreshed first for each Unit, followed by inputs from the Input Units to the CPU Unit.		I/O refresh time for each Unit multiplied by the number of Units used.
Special I/O Units	Words allocated in CIO Area		I/O refresh time for each Unit multiplied by the number of Units used.
	Unit-specific data	Example: CompoBus/S remote I/O	
CPU Bus Units	Words allocated in CIO and DM Areas		I/O refresh time for each Unit multiplied by the number of Units used.
	Unit-specific data	Examples: • EtherNet/IP tag data link refreshing • Controller Link data link refreshing • DeviceNet remote I/O communications • Send/receive data for protocol macros • Socket services for specific control bits for Ethernet Units	

5. Peripheral Servicing

The following peripheral device servicing is supported.

- Servicing for Communications Ports (Internal Logic Ports)
  Servicing for communications ports is performed when communications instructions in the ladder program or functions such as background execution or simple backup are used.
- Event Servicing for Special I/O Units or CPU Bus Units
  Event servicing is performed whenever a Unit is connected in the PLC. It is also performed in situations such as when Support Software is used via a Communications Unit.
- USB Port Servicing
  USB port servicing is performed when a USB port is being used.
- RS-232C Port Servicing
  RS-232C port servicing is performed when an RS-232C port is being used.
- File Access Servicing
  File access servicing is performed when a Memory Card or EM file memory is accessed.
- Online Edit Servicing
  Online edit servicing is performed when online editing is executed.
- Backup Servicing
  Backup servicing is performed when programs or parameters are changed.

When Peripheral Device servicing is not performed, no time is required for processing. The minimum and maximum processing times for Peripheral Device servicing depend on the Set Time to All Events parameter in the PLC Setup, as shown in the following table.

		Peripheral service execution time	
		Minimum	Maximum
Set Time to All Events parameter	**Disable**	0 ms (when servicing not executed)	10% of cycle time (or 0.1 ms if 10% of the cycle time is less than 0.1 ms)
	Enabled		Set time (0.1 to 3,276.7 ms)

12-2-3 I/O Unit Refresh Times for Individual Units

● Typical Basic I/O Unit Refresh Times

Name	Model	I/O refresh time per Unit	
		CJ2H	CJ2M
8/16-point DC Input Units	CJ1W-ID201/211/212	0.0014 ms	0.0039 ms
32-point DC Input Units	CJ1W-ID231/232/233	0.0023 ms	0.0054 ms
64-point DC Input Units	CJ1W-ID261/262	0.0041 ms	0.0089 ms
8/16-point AC Input Units	CJ1W-IA201/111	0.0014 ms	0.0039 ms
16-point Interrupt Input Units	CJ1W-INT01	0.0014 ms	0.0039 ms
16-point Quick-response Input Units	CJ1W-IDP01	0.0014 ms	0.0039 ms
8/16-point Relay Output Units	CJ1W-OC201/211	0.0014 ms	0.0039 ms
8-point Triac Output Units	CJ1W-OA201	0.0014 ms	0.0039 ms
8/16-point Transistor Output Units	CJ1W-OD201/202/203/204/211/212/213	0.0014 ms	0.0039 ms
32-point Transistor Output Units	CJ1W-OD231/232/233/234	0.0023 ms	0.0054 ms
64-point Transistor Output Units	CJ1W-OD261/262/263	0.0041 ms	0.0089 ms
24-VDC Input/Transistor Output Units (16 inputs/16 outputs)	CJ1W-MD231/232/233	0.0023 ms	0.0054 ms
24-VDC Input/Transistor Output Units (32 inputs/32 outputs)	CJ1W-MD261/263	0.0041 ms	0.0089 ms
TTL Input/TTL Output Units (16 inputs/16 outputs)	CJ1W-MD563	0.0041 ms	0.0089 ms
B7A Interface Unit (64 inputs)	CJ1W-B7A14	0.0041 ms	0.0089 ms
B7A Interface Unit (64 outputs)	CJ1W-B7A04	0.0041 ms	0.0089 ms
B7A Interface Unit (32 inputs/32 outputs)	CJ1W-B7A22	0.0041 ms	0.0089 ms

● Typical Special I/O Unit Refresh Times

Name	Model	I/O refresh time per Unit	
		CJ2H	CJ2M
Analog Input Units	CJ1W-AD041/081(V1)	0.05 ms	0.07 ms
	CJ1W-AD042	0.05 ms	0.07 ms
Analog Output Units	CJ1W-DA021/041/08V	0.05 ms	0.07 ms
	CJ1W-DA042V	0.05 ms	0.07 ms
Analog I/O Unit	CJ1W-MAD42	0.05 ms	0.07 ms
Temperature Control Units	CJ1W-TC□□□	0.05 ms	0.25 ms

Name	Model		I/O refresh time per Unit	
			CJ2H	CJ2M
Position Control Units	CJ1W-NC113/133		0.12 ms	0.13 ms
			+ 0.8 ms for each instruction (IOWR/IORD) used to transfer data.	
	CJ1W-NC213/233		0.14 ms	0.15 ms
			+ 0.8 ms for each instruction (IOWR/IORD) used to transfer data.	
	CJ1W-NC413/433		0.17 ms	0.18 ms
			+ 0.7 ms for each instruction (IOWR/ IORD) used to transfer data.	
	CJ1W-NC214/234 (2 axes and 2 tasks)		0.20 ms	0.355 ms
			+ 0.1 ms for each instruction (IOWR/IORD) used to transfer data.	
	CJ1W-NC414/434 (4 axes and 4 tasks)		0.30 ms	0.511 ms
			+ 0.1 ms for each instruction (IOWR/IORD) used to transfer data.	
ID Sensor Units	CJ1W-V600C11		0.12 ms	0.13 ms
	CJ1W-V600C12		0.14 ms	0.15 ms
High-speed Counter Unit	CJ1W-CT021		0.10 ms	0.11 ms
CompoNet Master Unit	CJ1W-CRM21	Communications mode No. 0	0.13 ms	0.14 ms
		Communications mode No. 1	0.14 ms	0.15 ms
		Communications mode No. 2	0.17 ms	0.17 ms
		Communications mode No. 3	0.21 ms	0.22 ms
		Communications mode No. 8 (*)	0.081 + (0.0012 × No. of words allocated) ms	0.083 + (0.009 × No. of words allocated) ms
CompoBus/S Master Unit	CJ1W-SRM21	Assigned 1 unit number	0.08 ms	0.08 ms
		Assigned 2 unit numbers	0.08 ms	0.10 ms

* The number of words allocated is the actually number of words in I/O memory actually allocated to the connected slaves.

Increase in Cycle Time Caused by CPU Bus Units

Name	Model	Increase		Remarks
		CJ2H	CJ2M	
Controller Link Unit	CJ1W-CLK21(V1)	0.1 ms	0.2 ms	The following additional time is required if data links are used. • CJ2H CPU Units: 0.1 ms + 0.7 μs × Number of data link words • CJ2M CPU Units: 0.1 ms + 1.2 μs × Number of data link words There will be an additional increase of the event execution times when message services are used.
Serial Communications Unit	CJ1W-SCU41-V1 CJ1W-SCU21-V1 CJ1W-SCU31-V1 CJ1W-SCU42 CJ1W-SCU32 CJ1W-SCU22	0.20 ms		There will be an increase of up to the following time when a protocol macro is executed: • CJ2H CPU Units: 0.4 μs × maximum number of data words sent or received (0 to 500 words) • CJ2M CPU Units: 0.6 μs × maximum number of data words sent or received (0 or more words) There will be an increase of the event execution times when Host Links or 1:N NT Links are used.
Ethernet Unit	CJ1W-ENT11/21	0.1 ms	0.9 ms	If socket services are executed with software switches, there will be an increase of up to the following time • CJ2H CPU Units: 1.4 μs × Number of bytes sent/received • CJ2M CPU Units: 0.17 μs × Number of bytes sent/received There will be an increase in the event execution times when FINS communications services, socket services for CMND instructions, or FTP services are performed.
EtherNet/IP Unit	CJ1W-EIP21	0.1 ms		Add the following value when tag data links are used. CJ2H CPU Units 0.1 ms + Number of words transferred × 0.33 μs* CJ2M CPU Units 0.1 ms + Number of words transferred × 0.70 μs* If the message service is used, the event execution time must also be added.
FL-net Unit	CJ1W-FLN22	0.25 ms + No. of data link words × 1.6 μs	0.25 ms + No. of data link words × 1.7 μs	The number of data link words is the number of words of data sent and received by a node. If the message service is used, the event execution time must be added separately.
DeviceNet Unit	CJ1W-DRM21	0.4 ms + 0.7 μs for each allocated word	0.4 ms + 0.4 μs for each allocated word	Include all words allocated to the slaves, including unused ones. For message communications, add the number of communications words to the calculations as the left.
Position Control Units with EtherCAT Communications	CJ1W-NC881 using 8 axes and 4 tasks	0.3	0.7	---
	CJ1W-NC481 using 4 axes and 4 tasks	0.3	0.6	
	CJ1W-NC281 using 2 axes and 2 tasks	0.2	0.5	

Name	Model	Increase		Remarks
		CJ2H	CJ2M	
Position Control Unit with MECHATROLINK-II Communications	CJ1W-NCF71	According to the number of Servo Driver axes connected to the Unit. 1 axis: 0.2 ms, 3 axes: 0.3 ms, 6 axes: 0.4 ms, 16 axes: 0.8 ms		---
Motion Control Unit with MECHATROLINK-II Communications	CJ1W-MCH71	0.2 + (No. of motion tasks × 4 + No. of axes × 5 + No. of words allocated for general I/O) × 0.001 (ms)	0.2 + (No. of motion tasks × 6 + No. of axes × 1 + No. of words allocated for general I/O) × 0.001 (ms)	---
SYSMAC SPU Unit	CJ1W-SPU01-V2	0.2 ms + No. of sampled words × 0.8 μs	0.2 ms + No. of sampled words × 1.1 μs	---

* The following value must be added when using high-speed interrupts with CJ2H CPU Units with unit version 1.1 or later.

0.1 ms + Number of words transferred × 0.87 μs

12-2-4 Cycle Time Calculation Example

● Example 1: Application Based on Basic Instructions and Basic I/O Units

The following example shows the method used to calculate the cycle time when only Basic I/O Units are connected in the PLC and the program consists of 20K steps of basic and data movement instructions. Here, a CJ2H-CPU6□-EIP CPU Unit is used.

Conditions

Item	Details	
CPU Unit	CJ2H-CPU6□-EIP	
CJ-series CPU Rack	CJ1W-ID261 64-point Input Units	2 Units
	CJ1W-OD261 64-point Output Units	2 Units
User program	20 Ksteps	LD instructions: 10 Ksteps MOV instructions: 10 Ksteps (LD: Each instruction is 1 step. MOV: Each instruction is 3 steps.)
Peripheral USB port connection	Yes and no	
Fixed cycle time processing	No	
Serial port connection	No	
Peripheral servicing with other devices (Special I/O Units, CPU Bus Units, and file access)	No	

Calculation Example

Process name	Calculation	Processing time	
		Programming Device connected to peripheral USB port	Programming Device not connected to peripheral USB port
(1) Overseeing	---	0.20 ms	0.20 ms
(2) Program execution	0.016 μs × 10,000 + 0.14 μs/3 steps × 10,000	0.63 ms	0.63 ms
(3) Cycle time calculation for minimum cycle time	(Fixed cycle time not set)	0 ms	0 ms
(4) I/O refreshing	0.0039 ms × 2 + 0.0039 ms × 2	0.0164 ms	0.0164 ms
(5) Peripheral servicing	(Peripheral USB port connection only)	0.1 ms	0 ms
Cycle time	(1) + (2) + (3) + (4) + (5)	0.95 ms	0.85 ms

● Example 2: Application Containing Calculations and Special I/O Units

The following example shows the method used to calculate the cycle time when Basic I/O Units and Special I/O Units are connected in the PLC and the program consists of 20K steps of basic instructions, data movement instructions, and floating-point calculation instructions. Here, a CJ2H-CPU6□-EIP CPU Unit is used.

Conditions

Item	Details	
CPU Unit	CJ2H-CPU6□-EIP	
CJ-series CPU Rack	CJ1W-ID261 64-point Input Units	2 Units
	CJ1W-OD261 64-point Output Units	2 Units
	CJ1W-AD081 Analog Input Unit	2 Units
	CJ1W-NC413 Position Control Unit	2 Units
User program	20 Ksteps	LD instructions: 12 Ksteps MOV instructions: 6 Ksteps +F instructions: 2K steps (LD: Each instruction is 1 step. MOV: Each instruction is 3 steps. +F: Each instruction is 4 steps)
Peripheral USB port connection	Yes and no	
Fixed cycle time processing	No	
Serial port connection	No	
Peripheral servicing with other devices (Special I/O Units, CPU Bus Units, and file access)	No	

Calculation Example

Process name	Calculation	Processing time	
		Programming Device connected to built-in USB port	Programming Device not connected to built-in USB port
(1) Overseeing	---	0.20 ms	0.20 ms
(2) Program execution	0.016 ms × 12,000 + 0.14 μs/3 steps × 6,000 + 0.24 μs/4 steps × 2,000	0.59 ms	0.59 ms
(3) Cycle time calculation for minimum cycle time	(Fixed cycle time not set)	0 ms	0 ms
(4) I/O refreshing	0.0041 ms × 2 + 0.0041 ms × 2 + 0.05 ms × 2 + 0.17 ms × 2	0.4564 ms	0.4564 ms
(5) Peripheral servicing	(Peripheral USB port connection only)	0.1 ms	0 ms
Cycle time	(1) + (2) + (3) + (4) + (5)	1.35 ms	1.25 ms

12-2-5 Online Editing Cycle Time Extension

Online editing can be executed from the CX-Programmer while the CPU Unit is in MONITOR mode. The cycle time will be extended by approximately 1 ms due to the write processing for the CPU Unit program. If you do not want the cycle time to be extended by a particular amount, use the online editing prohibit (standby) function. For details, refer to *10-7-4 Online Editing.*

12-2-6 I/O Response Time

The I/O response time is the time it takes from when an Input Unit's input turns ON, the data is recognized by the CJ-series CPU Unit, and the user program is executed, up to the time for the result to be output to an Output Unit's output terminals.

The length of the I/O response time depends on the following conditions.

- Timing of Input Bit turning ON.
- Cycle time.
- Type of Rack to which the Input and Output Units are mounted (CPU Rack or Expansion Rack).

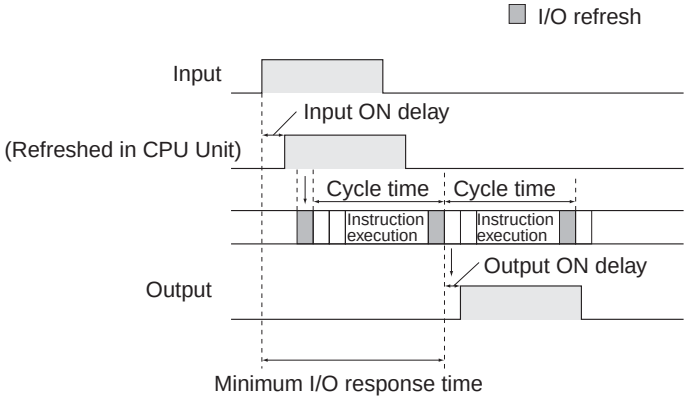
CJ-series Basic I/O Unit Response Times

● Minimum I/O Response Time

The I/O response time is shortest when data is retrieved immediately before I/O refresh of the CPU Unit.

The minimum I/O response time is the total of the Input ON delay, the cycle time, and the Output ON delay.

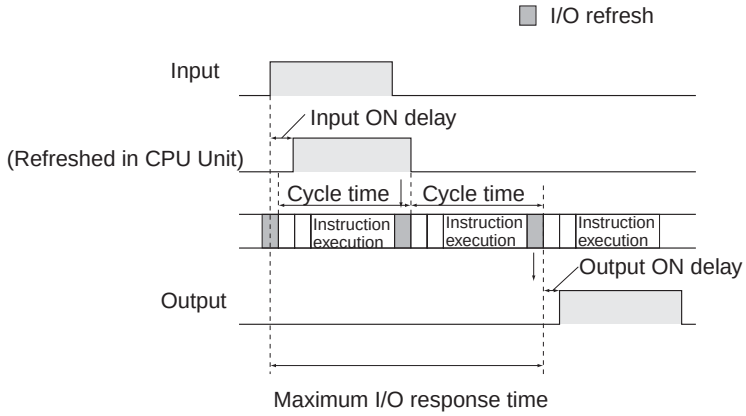
Note The Input and Output ON delay differs according to the Unit used.



● Maximum I/O Response Time

The I/O response time is longest when data is retrieved immediately after I/O refresh of the Input Unit.

The maximum I/O response time is the total of the Input ON delay, (the cycle time × 2), and the Output ON delay.



● Calculation Example

Conditions:

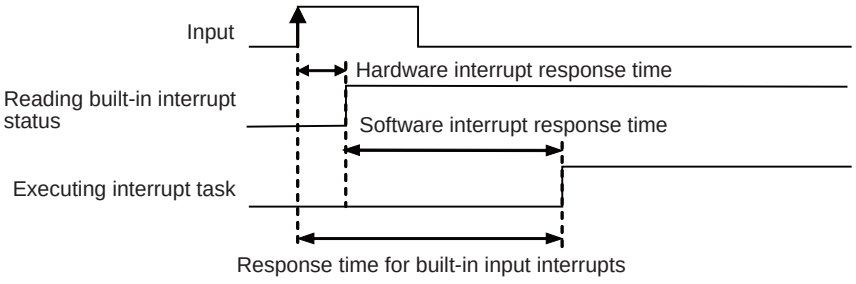
Input ON delay: 1.5 ms
Output ON delay: 0.2 ms
Cycle time: 20.0 ms

Minimum I/O response time = 1.5 ms + 20 ms + 0.2 ms = 21.7 ms

Maximum I/O response time = 1.5 ms + (20 ms × 2) + 0.2 ms = 41.7 ms

12-2-7 Response Time for Built-in Input Interrupts

The response time for built-in input interrupt tasks is the time from when a built-in input to a Pulse I/O Block connected to a CJ2M CPU Unit turns ON or OFF until the I/O interrupt task is executed. The length of the interrupt response time for built-in input interrupt tasks depends on the total of the hardware interrupt response time and software interrupt response time.



Response time for built-in input interrupts = Hardware interrupt response time + Software interrupt response time

12-2-8 Response Performance of Serial PLC Links

The response times for CPU Units connected via a Serial PLC Link (polling unit to polled unit or polled unit to polling unit) can be calculated as shown below.

- Maximum I/O response time (not including hardware delay) [ms] =
  Polling unit cycle time × 2 + Communications cycle time + Polled unit cycle time × 2 + Polled unit communications time + 4 ms
- Minimum I/O response time (not including hardware delay) [ms] =
  Polled unit communications time + 0.54 ms

Number of participating polled unit nodes	The number of polled units to which links have been established within the maximum unit number set in the polling unit.
Number of non-participating polled unit nodes	The number of polled units not participating in the links within the maximum unit number set in the polling unit.
Communications cycle time [ms]	• With Two or More Polled Units Polled unit communications time × Number of participating polled unit nodes + 10 × Number of non-participating polled unit nodes • With One Polled Unit Polled unit communications time + 10 ms
Polled unit communications time [ms]	• Baud rate set to Standard: Polled unit cycle time + 0.286 × [(Number of polled units + 1) × Number of link words × 2 + 12] • Baud rate set to Fast: Polled unit cycle time + 0.0955 × [(Number of polled units + 1) × Number of link words × 2 + 12]

App

Appendices

The appendices provide information on CPU Unit operation when power is interrupted, the instructions supported by the CPU Units, Auxiliary Area words and bits, instruction execution time, a memory map of the continuous PLC memory address, and a comparison of CJ-series and CS-series PLCs.

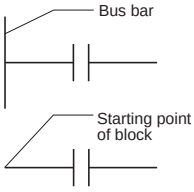
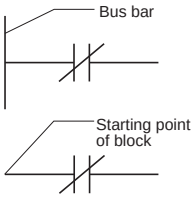
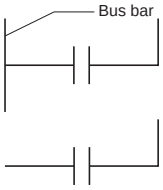
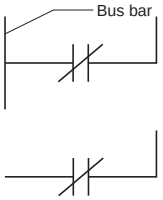
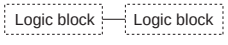
A-1 Instruction Functions A-3
A-1-1 Sequence Input Instructions A-3
A-1-2 Sequence Output Instructions A-5
A-1-3 Sequence Control Instructions A-6
A-1-4 Timer and Counter Instructions A-10
A-1-5 Comparison Instructions A-14
A-1-6 Data Movement Instructions A-18
A-1-7 Data Shift Instructions A-20
A-1-8 Increment/Decrement Instructions A-24
A-1-9 Symbol Math Instructions A-24
A-1-10 Conversion Instructions A-29
A-1-11 Logic Instructions A-35
A-1-12 Special Math Instructions A-37
A-1-13 Floating-point Math Instructions A-38
A-1-14 Double-precision Floating-point Instructions A-42
A-1-15 Table Data Processing Instructions A-45
A-1-16 Tracking Instructions A-49
A-1-17 Data Control Instructions A-50
A-1-18 Subroutine Instructions A-54
A-1-19 Interrupt Control Instructions A-55
A-1-20 Step Instructions A-56
A-1-21 Basic I/O Unit Instructions A-56
A-1-22 Serial Communications Instructions A-59
A-1-23 Network Instructions A-61
A-1-24 File Memory Instructions A-63
A-1-25 Display Instructions A-64
A-1-26 Clock Instructions A-65
A-1-27 Debugging Instructions A-66
A-1-28 Failure Diagnosis Instructions A-66
A-1-29 Other Instructions A-67
A-1-30 Block Programming Instructions A-68
A-1-31 Text String Processing Instructions A-72
A-1-32 Task Control Instructions A-75
A-1-33 Model Conversion Instructions A-75

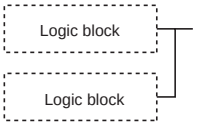
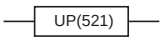
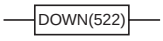
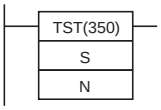
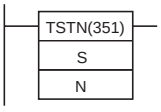
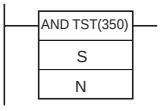
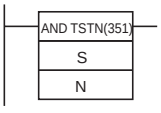
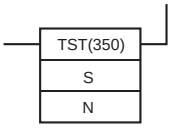
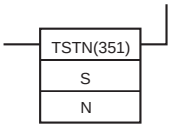
A-1-34 Special Function Block Instructions A-76
A-2 Instruction Execution Times and Number of Steps A-78
A-2-1 Sequence Input Instructions A-79
A-2-2 Sequence Output Instructions A-79
A-2-3 Sequence Control Instructions A-80
A-2-4 Timer and Counter Instructions A-81
A-2-5 Comparison Instructions A-82
A-2-6 Data Movement Instructions A-83
A-2-7 Data Shift Instructions A-84
A-2-8 Increment/Decrement Instructions A-85
A-2-9 Symbol Math Instructions A-85
A-2-10 Conversion Instructions A-87
A-2-11 Logic Instructions A-89
A-2-12 Special Math Instructions A-89
A-2-13 Floating-point Math Instructions A-89
A-2-14 Double-precision Floating-point Instructions A-91
A-2-15 Table Data Processing Instructions A-92
A-2-16 Tracking Instructions A-94
A-2-17 Data Control Instructions A-94
A-2-18 Subroutine Instructions A-95
A-2-19 Interrupt Control Instructions A-95
A-2-20 Step Instructions A-96
A-2-21 Basic I/O Unit Instructions A-96
A-2-22 Serial Communications Instructions A-97
A-2-23 Network Instructions A-98
A-2-24 File Memory Instructions A-98
A-2-25 Display Instructions A-98
A-2-26 Clock Instructions A-98
A-2-27 Debugging Instructions A-99
A-2-28 Failure Diagnosis Instructions A-99
A-2-29 Other Instructions A-100
A-2-30 Block Programming Instructions A-100
A-2-31 Text String Processing Instructions A-102
A-2-32 Task Control Instructions A-103
A-2-33 Model Conversion Instructions A-103
A-2-34 Special Function Block Instructions A-103
A-2-35 SFC Instructions A-103
A-2-36 Function Block Instance Execution Time A-104
A-3 Auxiliary Area A-106
A-3-1 Read-only Area (Set by System) A-106
A-3-2 Read/Write Area (Set by User) A-129
A-3-3 Details on Auxiliary Area Operation A-138
A-4 Memory Map of PLC Memory Addresses A-141
A-4-1 PLC Memory Addresses A-141
A-4-2 Memory Map A-142
A-5 Operation for Power Interruptions A-143
A-5-1 Power OFF Operation A-143
A-5-2 Instruction Execution for Power Interruptions A-145
A-6 EtherNet/IP Connections from Windows XP (SP2 or Higher) or Windows Vista A-147
A-6-1 Changing Windows Firewall Settings A-147
A-7 PLC Comparison Charts: CJ-series and CS-series PLCs A-150
A-8 Functions Supported for Unit Versions A-154

A-1 Instruction Functions

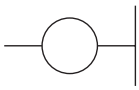
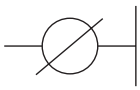
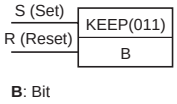
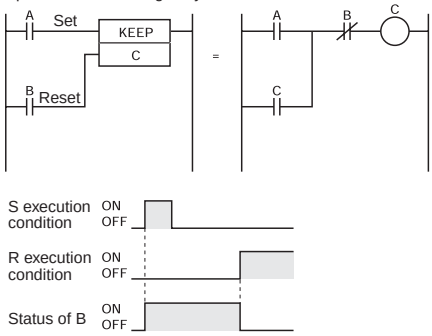
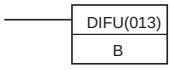
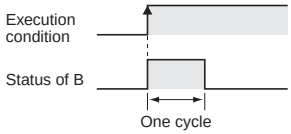
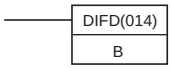
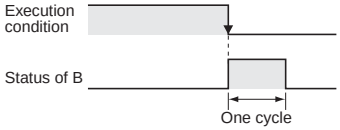
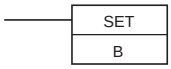
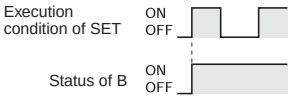
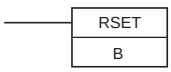
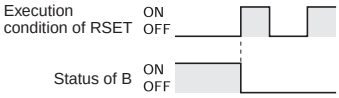
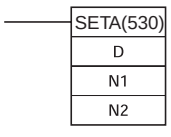
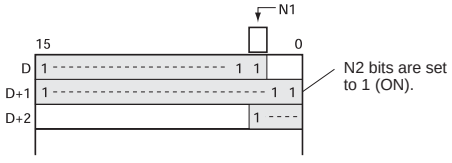
The CJ2 CPU Units support the following instructions.
Refer to the *CS/CJ/NSJ-series Instructions Reference Manual* (Cat. No. W474) for details.

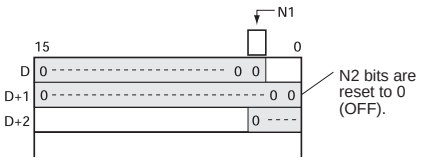
A-1-1 Sequence Input Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LOAD	LD @LD %LD !LD !@LD !%LD	---		Indicates a logical start and creates an ON/OFF execution condition based on the ON/OFF status of the specified operand bit.	Logic start	Not required
LOAD NOT	LD NOT @LD NOT*2 %LD NOT !LD NOT !@LD NOT !%LD NOT	---		Indicates a logical start and creates an ON/OFF execution condition based on the reverse of the ON/OFF status of the specified operand bit.	Logic start	Not required
AND	AND @AND %AND !AND !@AND !%AND	---	Symbol 	Takes a logical AND of the status of the specified operand bit and the current execution condition.	Continues on rung	Required
AND NOT	AND NOT @AND NOT %AND NOT !AND NOT !@AND NOT !%AND NOT	---	Symbol 	Reverses the status of the specified operand bit and takes a logical AND with the current execution condition.	Continues on rung	Required
OR	OR @OR %OR !OR !@OR !%OR	---		Takes a logical OR of the ON/OFF status of the specified operand bit and the current execution condition.	Continues on rung	Required
OR NOT	OR NOT @OR NOT %OR NOT !OR NOT !@OR NOT !%OR NOT	---		Reverses the status of the specified bit and takes a logical OR with the current execution condition	Continues on rung	Required
AND LOAD	AND LD	---		Takes a logical AND between logic blocks. LD to … Logic block A LD to … Logic block B AND LD ····· Serial connection between logic block A and logic block B.	Continues on rung	Required

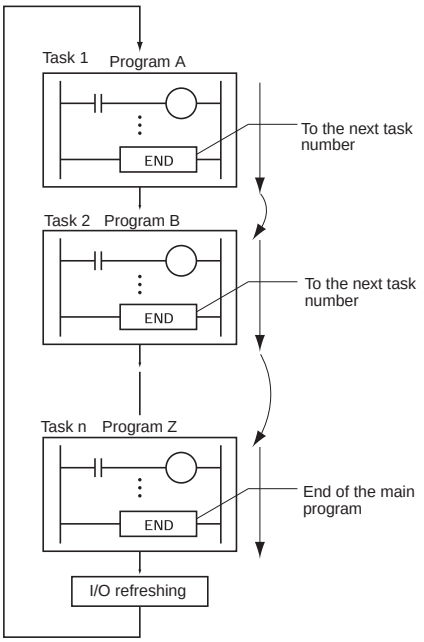
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
OR LOAD	OR LD	---		Takes a logical OR between logic blocks. LD to … Logic block A LD to … Logic block B OR LD ····· Parallel connection between logic block A and logic block B.	Continues on rung	Required
NOT	NOT	520		Reverses the execution condition.	Continues on rung	Required
CONDITION ON	UP	521		UP(521) turns ON the execution condition for one cycle when the execution condition goes from OFF to ON.	Continues on rung	Required
CONDITION OFF	DOWN	522		DOWN(522) turns ON the execution condition for one cycle when the execution condition goes from ON to OFF.	Continues on rung	Required
BIT TEST	LD TST	350	 **S**: Source word **N**: Bit number	LD TST(350), AND TST(350), and OR TST(350) are used in the program like LD, AND, and OR; the execution condition is ON when the specified bit in the specified word is ON and OFF when the bit is OFF.	Logic start	Not required
BIT TEST	LD TSTN	351	 **S**: Source word **N**: Bit number	LD TSTN(351), AND TSTN(351), and OR TSTN(351) are used in the program like LD NOT, AND NOT, and OR NOT; the execution condition is OFF when the specified bit in the specified word is ON and ON when the bit is OFF.	Logic start	Not required
BIT TEST	AND TST	350	 **S**: Source word **N**: Bit number	LD TST(350), AND TST(350), and OR TST(350) are used in the program like LD, AND, and OR; the execution condition is ON when the specified bit in the specified word is ON and OFF when the bit is OFF.	Continues on rung	Required
BIT TEST	AND TSTN	351	 **S**: Source word **N**: Bit number	LD TSTN(351), AND TSTN(351), and OR TSTN(351) are used in the program like LD NOT, AND NOT, and OR NOT; the execution condition is OFF when the specified bit in the specified word is ON and ON when the bit is OFF.	Continues on rung	Required
BIT TEST	OR TST	350	 **S**: Source word **N**: Bit number	LD TST(350), AND TST(350), and OR TST(350) are used in the program like LD, AND, and OR; the execution condition is ON when the specified bit in the specified word is ON and OFF when the bit is OFF.	Continues on rung	Required
BIT TEST	OR TSTN	351	 **S**: Source word **N**: Bit number	LD TSTN(351), AND TSTN(351), and OR TSTN(351) are used in the program like LD NOT, AND NOT, and OR NOT; the execution condition is OFF when the specified bit in the specified word is ON and ON when the bit is OFF.	Continues on rung	Required

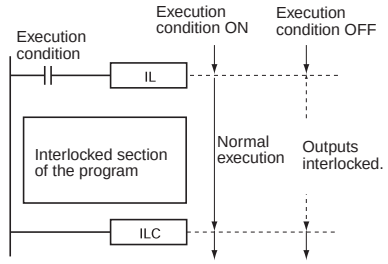
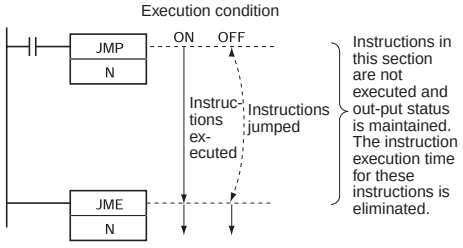
A-1-2 Sequence Output Instructions

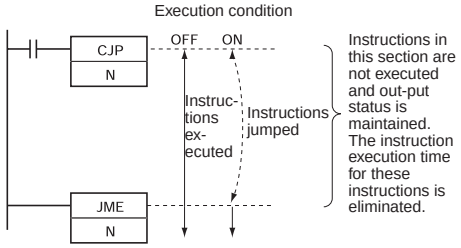
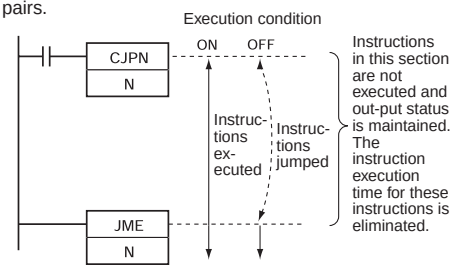
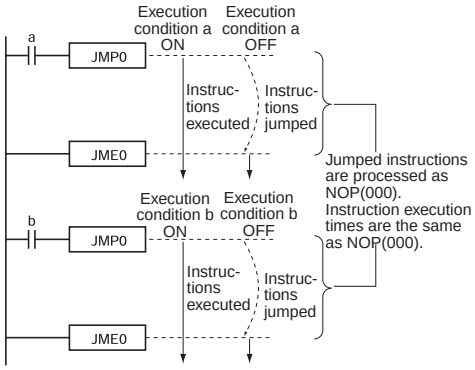
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
OUTPUT	OUT !OUT	---		Outputs the result (execution condition) of the logical processing to the specified bit.	Output	Required
OUTPUT NOT	OUT NOT !OUT NOT	---		Reverses the result (execution condition) of the logical processing, and outputs it to the specified bit.	Output	Required
KEEP	KEEP !KEEP	011	 **B**: Bit	Operates as a latching relay. 	Output	Required
DIFFERENTIATE UP	DIFU !DIFU	013	 **B**: Bit	DIFU(013) turns the designated bit ON for one cycle when the execution condition goes from OFF to ON (rising edge). 	Output	Required
DIFFERENTIATE DOWN	DIFD !DIFD	014	 **B**: Bit	DIFD(014) turns the designated bit ON for one cycle when the execution condition goes from ON to OFF (falling edge). 	Output	Required
SET	SET @SET %SET !SET !@SET !%SET	---	 **B**: Bit	SET turns the operand bit ON when the execution condition is ON. 	Output	Required
RESET	RSET @RSET %RSET !RSET !@RSET !%RSET	---	 **B**: Bit	RSET turns the operand bit OFF when the execution condition is ON. 	Output	Required
MULTIPLE BIT SET	SETA @SETA	530	 **D**: Beginning word **N1**: Beginning bit **N2**: Number of bits	SETA(530) turns ON the specified number of consecutive bits. 	Output	Required

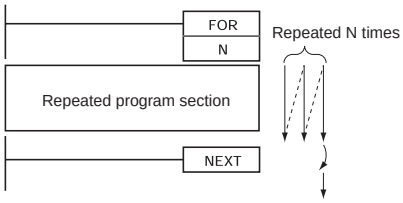
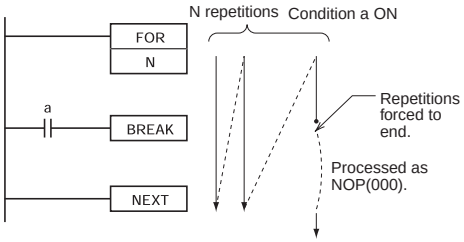
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MULTIPLE BIT RESET	RSTA @RSTA	531	RSTA(531) D N1 N2 **D**: Beginning word **N1**: Beginning bit **N2**: Number of bits	RSTA(531) turns OFF the specified number of consecutive bits. 	Output	Required
SINGLE BIT SET	SETB @SETB !SETB !@SETB	532	SETB(532) D N **D**: Word address **N**: Bit number	SETB(532) turns ON the specified bit in the specified word when the execution condition is ON. Unlike the SET instruction, SETB(532) can be used to set a bit in a DM or EM word.	SINGLE BIT SET	SETB
SINGLE BIT RESET	RSTB @RSTB !RSTB !@RSTB	533	RSTB(533) D N **D**: Word address **N**: Bit number	RSTB(533) turns OFF the specified bit in the specified word when the execution condition is ON. Unlike the RSET instruction, RSTB(533) can be used to reset a bit in a DM or EM word.	Output	Required
SINGLE BIT OUTPUT	OUTB @OUTB !OUTB	534	OUTB(534) D N **D**: Word address **N**: Bit number	OUTB(534) outputs the result (execution condition) of the logical processing to the specified bit. Unlike the OUT instruction, OUTB(534) can be used to control a bit in a DM or EM word.	Output	Required

A-1-3 Sequence Control Instructions

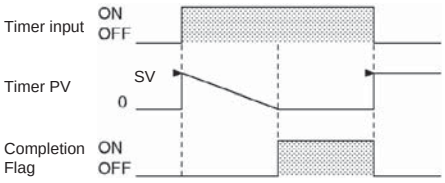
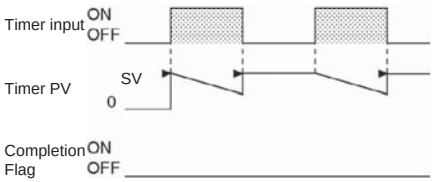
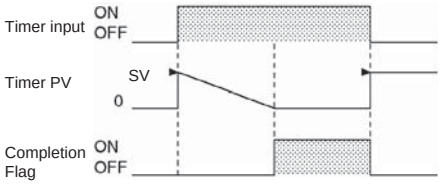
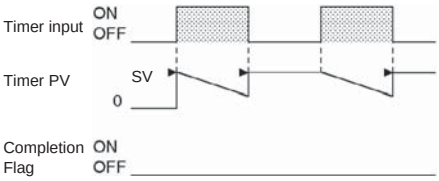
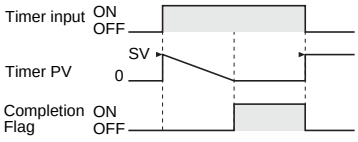
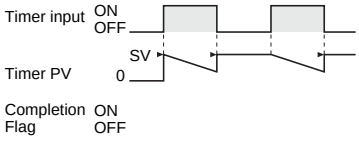
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
END	END	001	END(001)	Indicates the end of a program. END(001) completes the execution of a program for that cycle. No instructions written after END(001) will be executed. Execution proceeds to the program with the next task number. When the program being executed has the highest task number in the program, END(001) marks the end of the overall main program. 	Output	Not required
NO OPERATION	NOP	000	---	This instruction has no function. (No processing is performed for NOP(000).)	Output	Not required

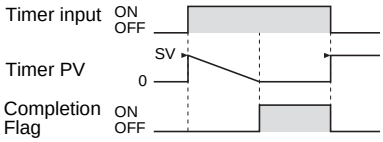
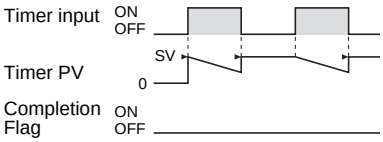
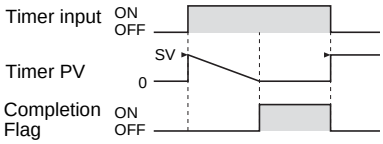
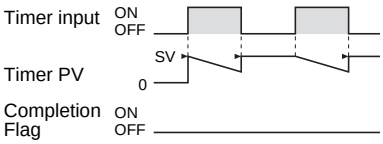
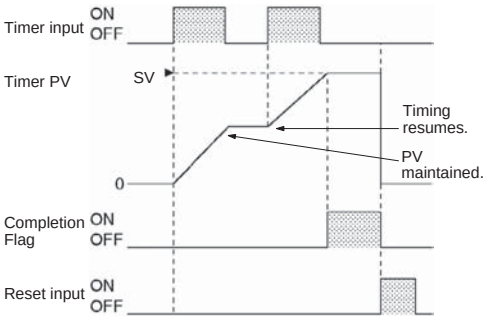
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
INTERLOCK	IL	002	IL(002)	Interlocks all outputs between IL(002) and ILC(003) when the execution condition for IL(002) is OFF. IL(002) and ILC(003) are normally used in pairs. 	Output	Required
INTERLOCK CLEAR	ILC	003	ILC(003)	All outputs between IL(002) and ILC(003) are interlocked when the execution condition for IL(002) is OFF. IL(002) and ILC(003) are normally used in pairs.	Output	Not required
MULTI-INTER-LOCK DIFFER-ENTIATION HOLD	MILH	517	MILH (517) N D **N**: Interlock number **D**: Interlock Status Bit	When the execution condition for MILH(517) is OFF, the outputs for all instructions between that MILH(517) instruction and the next MILC(519) instruction are interlocked. MILH(517) and MILC(519) are used as a pair. MILH(517)/MILC(519) interlocks can be nested (e.g., MILH(517)-MILH(517)-MILC(519)-MILC(519)). If there is a differentiated instruction (DIFU, DIFD, or instruction with a @ or % prefix) between MILH(517) and the corresponding MILC(519), that instruction will be executed after the interlock is cleared if the differentiation condition of the instruction was established.	Output	Required
MULTI-INTER-LOCK DIFFER-ENTIATION RELEASE	MILR	518	MILR (518) N D **N**: Interlock number **D**: Interlock Status Bit	When the execution condition for MILR(518) is OFF, the outputs for all instructions between that MILR(518) instruction and the next MILC(519) instruction are interlocked.MILR(518) and MILC(519) are used as a pair. MILR(518)/MILC(519) interlocks can be nested (e.g., MILR(518)-MILR(518)-MILC(519)-MILC(519)). If there is a differentiated instruction (DIFU, DIFD, or instruction with a @ or % prefix) between MILR(518) and the corresponding MILC(519), that instruction will not be executed after the interlock is cleared even if the differentiation condition of the instruction was established.	Output	Required
MULTI-INTER-LOCK CLEAR	MILC	519	MILC (519) N **N**: Interlock number	Clears an interlock started by an MILH(517) or MILR(518) with the same interlock number. All outputs between MILH(517)/MILR(518) and the corresponding MILC(519) with the same interlock number are interlocked when the execution condition for MILH(517)/MILR(518) is OFF.	Output	Not required
JUMP	JMP	004	JMP(004) N **N**: Jump number	When the execution condition for JMP(004) is OFF, program execution jumps directly to the first JME(005) in the program with the same jump number. JMP(004) and JME(005) are used in pairs. 	Output	Required

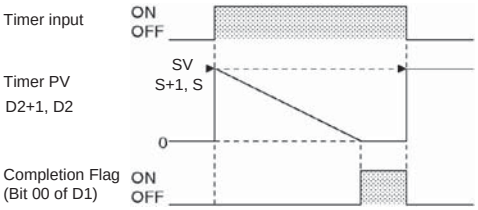
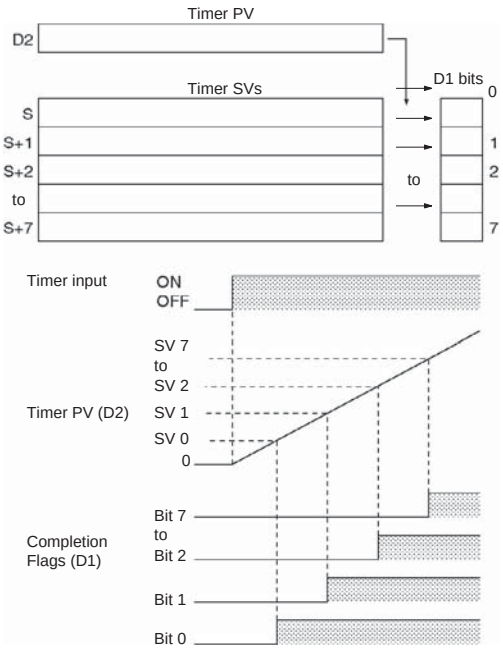
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CONDITIONAL JUMP	CJP	510	CJP(510) N **N**: Jump number	The operation of CJP(510) is the basically the opposite of JMP(004). When the execution condition for CJP(510) is ON, program execution jumps directly to the first JME(005) in the program with the same jump number. CJP(510) and JME(005) are used in pairs. 	Output	Required
CONDITIONAL JUMP	CJPN	511	CJPN(511) N **N**: Jump number	The operation of CJPN(511) is almost identical to JMP(004). When the execution condition for CJP(004) is OFF, program execution jumps directly to the first JME(005) in the program with the same jump number. CJPN(511) and JME(005) are used in pairs. 	Output	Not required
JUMP END	JME	005	JME(005) N **N**: Jump number	Indicates the destination of a jump instruction.	Output	Not required
MULTIPLE JUMP	JMP0	515	JMP0(515)	When the execution condition for JMP0(515) is OFF, all instructions from JMP0(515) to the next JME0(516) in the program are processed as NOP(000). Use JMP0(515) and JME0(516) in pairs. There is no limit on the number of pairs that can be used in the program. 	Output	Required
MULTIPLE JUMP END	JME0	516	JME0(516)	When the execution condition for JMP0(515) is OFF, all instructions from JMP0(515) to the next JME0(516) in the program are processed as NOP(000). Use JMP0(515) and JME0(516) in pairs. There is no limit on the number of pairs that can be used in the program.	Output	Not required

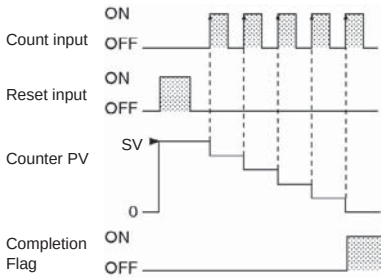
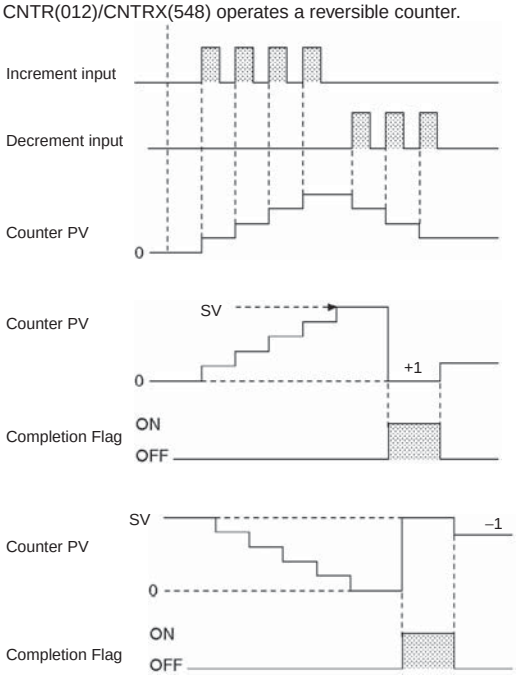
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FOR-NEXT LOOPS	FOR	512	FOR(512) N **N**: Number of loops	The instructions between FOR(512) and NEXT(513) are repeated a specified number of times. FOR(512) and NEXT(513) are used in pairs. 	Output	Not required
BREAK LOOP	BREAK	514	BREAK(514)	Programmed in a FOR-NEXT loop to cancel the execution of the loop for a given execution condition. The remaining instructions in the loop are processed as NOP(000) instructions. 	Output	Required
FOR-NEXT LOOPS	NEXT	513	NEXT(513)	The instructions between FOR(512) and NEXT(513) are repeated a specified number of times. FOR(512) and NEXT(513) are used in pairs.	Output	Not required

A-1-4 Timer and Counter Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
HUNDRED-MS TIMER	TIM (BCD)	---	TIM N S **N**: Timer number **S**: Set value	TIM/TIMX(550) operates a decrementing timer with units of 0.1-s. The setting range for the set value (SV) is 0 to 999.9 s for BCD and 0 to 6,553.5 s for binary (decimal or hexadecimal).  Timer Input Turned OFF before Completion Flag Turns ON 	Output	Required
	TIMX (Binary)	550	TIMX(550) N S **N**: Timer number **S**: Set value			
TEN-MS TIMER	TIMH (BCD)	015	TIMH(015) N S **N**: Timer number **S**: Set value	TIMH(015)/TIMHX(551) operates a decrementing timer with units of 10-ms. The setting range for the set value (SV) is 0 to 99.99 s for BCD and 0 to 655.35 s for binary (decimal or hexadecimal).  Timer Input Turned OFF before Completion Flag Turns ON 	Output	Required
	TIMHX (Binary)	551	TIMHX(551) N S **N**: Timer number **S**: Set value			
ONE-MS TIMER	TMHH (BCD)	540	TMHH(540) N S **N**: Timer number **S**: Set value	TMHH(540)/TMHHX(552) operates a decrementing timer with units of 1-ms. The setting range for the set value (SV) is 0 to 9.999 s for BCD and 0 to 65.535 s for binary (decimal or hexadecimal). The timing charts for TMHH(540) and TMHHX(552) are the same as those given above for TIMH(015).  Timer Input Turned OFF before Completion Flag Turns ON 	Output	Required
	TMHHX (BCD)	552	TMHHX(552) N S **N**: Timer number **S**: Set value			

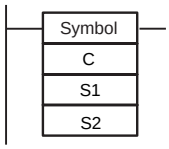
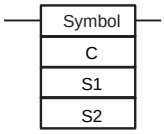
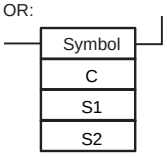
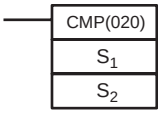
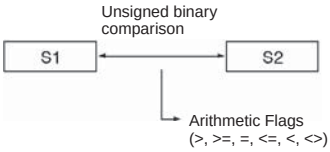
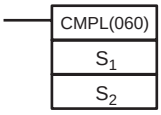
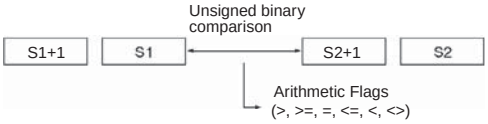
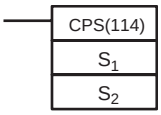
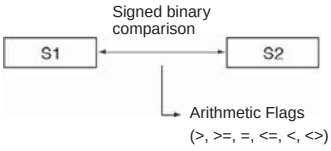
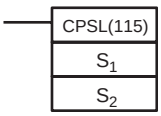
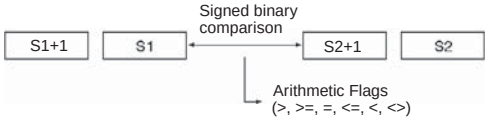
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
TENTH-MS TIMER	TIMU (BCD)	541	TIMU (541) N S **N**: Timer number **S**: Set value	TIMU(541)/TIMUX(556) operates a decrementing timer with units of 0.1-ms. The setting range for the set value (SV) is 0 to 0.9999 s for BCD and 0 to 6.5535 s for binary (decimal or hexadecimal).  Timer Input Turned OFF before Completion Flag Turns ON 	Output	Required
	TIMUX (BIN)	556	TIMUX(556) N S **N**: Timer number **S**: Set value			
HUNDREDTH-MS TIMER	TMUH (BCD)	544	TMUH(544) N S **N**: Timer number **S**: Set value	TMUH(544)/TMUHX(557) operates a decrementing timer with units of 0.01-ms. The setting range for the set value (SV) is 0 to 0.09999 s for BCD and 0 to 0.65535 s for binary (decimal or hexadecimal).  Timer Input Turned OFF before Completion Flag Turns ON 	Output	Required
	TMUX (BIN)	556	TMUHX(557) N S **N**: Timer number **S**: Set value			
ACCUMULA-TIVE TIMER	TTIM (BCD)	087	Timer input TTIM(087) N S Reset input **N**: Timer number **S**: Set value	TTIM(087)/TTIMX(555) operates an incrementing timer with units of 0.1-s. The setting range for the set value (SV) is 0 to 999.9 s for BCD and 0 to 6,553.5 s for binary (decimal or hexadecimal). 	Output	Required
	TTIMX (Binary)	555	Timer input TTIMX(555) N S Reset input **N**: Timer number **S**: Set value			

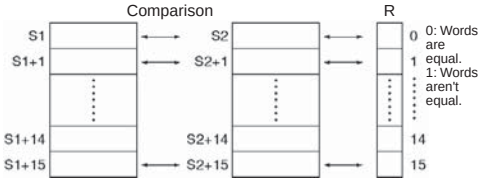
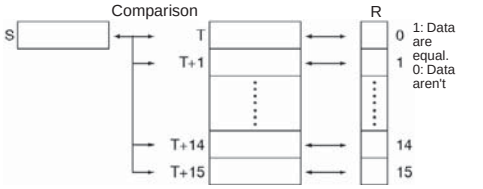
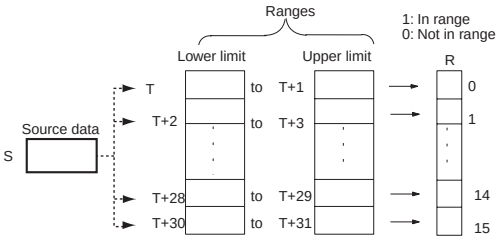
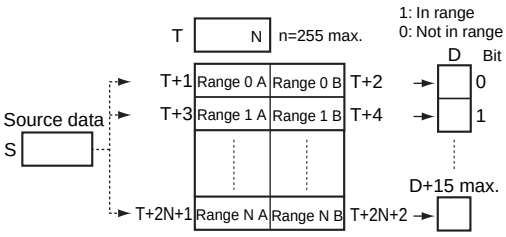
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LONG TIMER	TIML (BCD)	542	TIML(542) D1 D2 S **D1**: Completion Flag **D2**: PV word **S**: SV word	TIML(542)/TIMLX(553) operates a decrementing timer with units of 0.1-s that can time up to approx. 115 days for BCD and 49,710 days for binary (decimal or hexadecimal). 	Output	Required
	TIMLX (Binary)	553	TIMLX(553) D1 D2 S **D1**: Completion Flag **D2**: PV word **S**: SV word			
MULTI-OUTPUT TIMER	MTIM (BCD)	543	MTIM(543) D1 D2 S **D1**: Completion Flags **D2**: PV word **S**: 1st SV word	MTIM(543)/MTIMX(554) operates a 0.1-s incrementing timer with 8 independent SVs and Completion Flags. The setting range for the set value (SV) is 0 to 999.9 s for BCD and 0 to 6,553.5 s for binary (decimal or hexadecimal). 	Output	Required
	MTIMX (Binary)	554	MTIMX(554) D1 D2 S **D1**: Completion Flags **D2**: PV word **S**: 1st SV word			
TIMER RESET	TRSET	549	TRSET(549) N **N**: Timer number	Resets the specified timer.	Output	Required

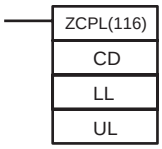
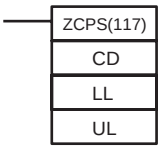
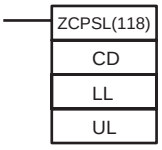
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
COUNTER	CNT (BCD)	---	Count input, CNT, N, S, Reset input **N**: Counter number **S**: Set value	CNT/CNTX(546) operates a decrementing counter. The setting range for the set value (SV) is 0 to 9,999 for BCD and 0 to 65,535 for binary (decimal or hexadecimal). 	Output	Required
	CNTX (Binary)	546	Count input, CNTX(546), N, S, Reset input **N**: Counter number **S**: Set value			
REVERSIBLE COUNTER	CNTR (BCD)	012	Increment input, Decrement input, Reset input, CNTR(012), N, S **N**: Counter number **S**: Set value	CNTR(012)/CNTRX(548) operates a reversible counter. 	Output	Required
	CNTRX (Binary)	548	Increment input, Decrement input, Reset input, CNTRX(548), N, S **N**: Counter number **S**: Set value			
RESET TIMER/ COUNTER	CNR @CNR (BCD)	545	CNR(545), N1, N2 **N1**: 1st number in range **N2**: Last number in range	CNR(545)/CNRX(547) resets the timers or counters within the specified range of timer or counter numbers. Sets the set value (SV) to the maximum of #9999 for BCD and #FFFF for binary.	Output	Required
	CNRX @CNRX (Binary)	547	CNRX(547), N1, N2 **N1**: 1st number in range **N2**: Last number in range			

A-1-5 Comparison Instructions

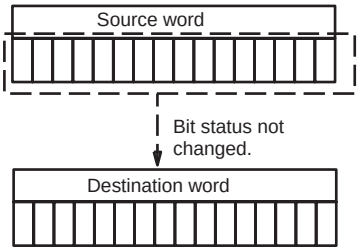
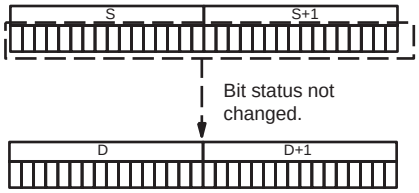
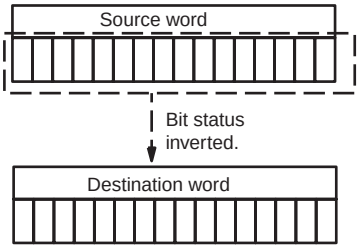
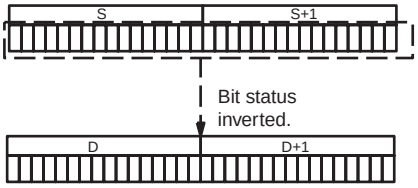
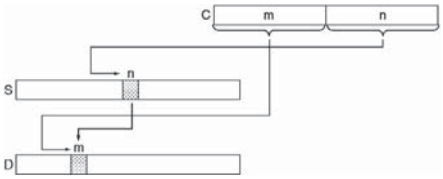
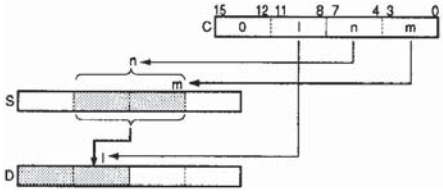
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
Symbol Comparison (Unsigned)	LD, AND, OR + =, <>, <, <=, >, >=	300 (=) 305 (<>) 310 (<) 315 (<=) 320 (>) 325(>=)	LD Symbol and option S_1 S_2 AND Symbol and option S_1 S_2 OR Symbol and option S_1 S_2 **S1**: Comparison data 1 **S2**: Comparison data 2	Symbol comparison instructions (unsigned) compare two values (constants and/or the contents of specified words) in 16-bit binary data and create an ON execution condition when the comparison condition is true. There are three types of symbol comparison instructions, LD (LOAD), AND, and OR. LD ON execution condition when comparison result is true. AND ON execution condition when comparison result is true. OR ON execution condition when comparison result is true.	LD: Logic start AND, OR: Continues on rung	LD: Not required AND, OR: Required
Symbol Comparison (Double-word, unsigned)	LD, AND, OR + =, <>, <, <=, >, >= + L	301 (=) 306 (<>) 311 (<) 316 (<=) 321 (>) 326 (>=)	S_1: Comparison data 1 S_2: Comparison data 2	Symbol comparison instructions (double-word, unsigned) compare two values (constants and/or the contents of specified double-word data) in unsigned 32-bit binary data and create an ON execution condition when the comparison condition is true. There are three types of symbol comparison instructions, LD (LOAD), AND, and OR.	LD: Logic start AND, OR: Continues on rung	LD: Not required AND, OR: Required
Symbol Comparison (Signed)	LD, AND, OR + =, <>, <, <=, >, >= +S	302 (=) 307 (<>) 312 (<) 317 (<=) 322 (>) 327 (>=)	S_1: Comparison data 1 S_2: Comparison data 2	Symbol comparison instructions (signed) compare two values (constants and/or the contents of specified words) in signed 16-bit binary (4-digit hexadecimal) and create an ON execution condition when the comparison condition is true. There are three types of symbol comparison instructions, LD (LOAD), AND, and OR.	LD: Logic start AND, OR: Continues on rung	LD: Not required AND, OR: Required
Symbol Comparison (Double-word, signed)	LD, AND, OR + =, <>, <, <=, >, >= +SL	303 (=) 308 (<>) 313 (<) 318 (<=) 323 (>) 328 (>=)	S_1: Comparison data 1 S_2: Comparison data 2	Symbol comparison instructions (double-word, signed) compare two values (constants and/or the contents of specified double-word data) in signed 32-bit binary (8-digit hexadecimal) and create an ON execution condition when the comparison condition is true. There are three types of symbol comparison instructions, LD (LOAD), AND, and OR.	LD: Logic start AND, OR: Continues on rung	LD: Not required AND, OR: Required

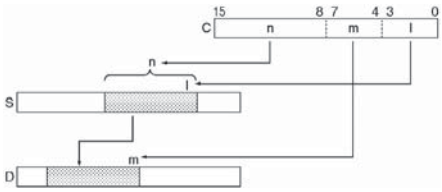
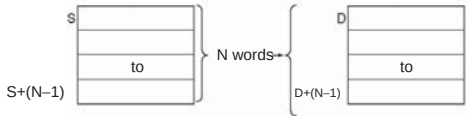
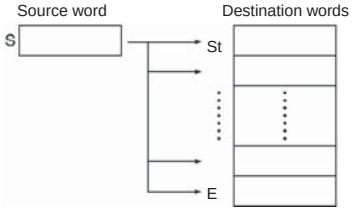
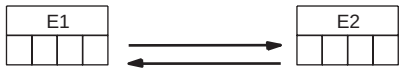
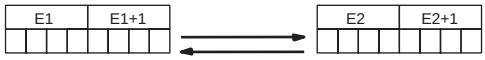
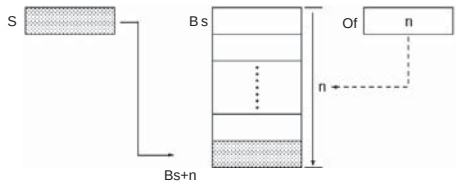
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
Time Comparison	LD, AND, OR + = DT, <> DT, < DT, <= DT, > DT, >= DT	341 (= DT) 342 (<> DT) 343 (< DT) 344 (<= DT) 345 (> DT) 346 (>= DT)	LD (LOAD):  AND:  OR:  **C:** Control word **S1:** 1st word of present time **S2:** 1st word of comparison time	Time comparison instructions compare two BCD time values and create an ON execution condition when the comparison condition is true. There are three types of time comparison instructions, LD (LOAD), AND, and OR. Time values (year, month, day, hour, minute, and second) can be masked/unmasked in the comparison so it is easy to create calendar timer functions.	LD: Logic start AND, OR: Continues on rung	LD: Not required AND, OR: Required
UNSIGNED COMPARE	CMP !CMP	020	 **S1**: Comparison data 1 **S2**: Comparison	Compares two unsigned binary values (constants and/or the contents of specified words) and outputs the result to the Arithmetic Flags in the Auxiliary Area. 	Output	Required
DOUBLE UNSIGNED COMPARE	CMPL	060	 **S1**: Comparison data 1 **S2**: Comparison data 2	Compares two double unsigned binary values (constants and/or the contents of specified words) and outputs the result to the Arithmetic Flags in the Auxiliary Area. 	Output	Required
SIGNED BINARY COMPARE	CPS !CPS	114	 **S1**: Comparison data 1 **S2**: Comparison data 2	Compares two signed binary values (constants and/or the contents of specified words) and outputs the result to the Arithmetic Flags in the Auxiliary Area. 	Output	Required
DOUBLE SIGNED BINARY COMPARE	CPSL	115	 **S1**: Comparison data 1 **S2**: Comparison	Compares two double signed binary values (constants and/or the contents of specified words) and outputs the result to the Arithmetic Flags in the Auxiliary Area. 	Output	Required

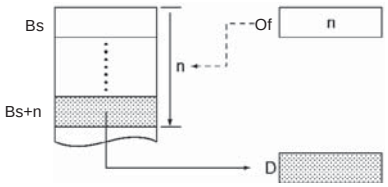
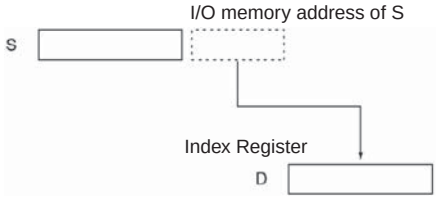
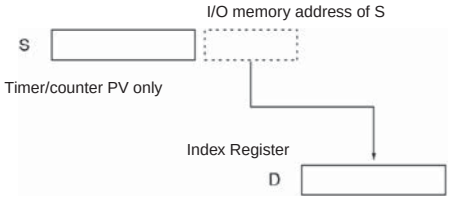
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MULTIPLE COMPARE	MCMP @MCMP	019	MCMP(019) S_1 S_2 R **S_1**: 1st word of set 1 **S_2**: 1st word of set 2 **R**: Result word	Compares 16 consecutive words with another 16 consecutive words and turns ON the corresponding bit in the result word where the contents of the words **are not** equal. 	Output	Required
TABLE COMPARE	TCMP @TCMP	085	TCMP(085) S T R **S**: Source data **T**: 1st word of table **R**: Result word	Compares the source data to the contents of 16 words and turns ON the corresponding bit in the result word when the contents are equal. 	Output	Required
UNSIGNED BLOCK COMPARE	BCMP @BCMP	068	BCMP(068) S T R **S**: Source data **T**: 1st word of table **R**: Result word	Compares the source data to 16 ranges (defined by 16 lower limits and 16 upper limits) and turns ON the corresponding bit in the result word when the source data is within the range. 	Output	Required
EXPANDED BLOCK COMPARE	BCMP2 @BCMP2	502	BCMP2(502) S T R **S**: Source data **T**: 1st word of block **R**: Result word	Compares the source data to up to 256 ranges (defined by upper and lower limits) and turns ON the corresponding bit in the result word when the source data is within a range.  **Note:** A can be less than or equal to B or greater the B.	Output	Required
AREA RANGE COMPARE	ZCP	088	ZCP(088) CD LL UL **CD**: Compare data (1 word) **LL**: Lower limit of range **UL**: Upper limit of range	Compares the 16-bit unsigned binary value in CD (word contents or constant) to the range defined by LL and UL and outputs the results to the Arithmetic Flags in the Auxiliary Area.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE AREA RANGE COMPARE	ZCPL	116	 **CD**: Compare data (2 words) **LL**: First word of lower limit **UL**: First word of upper limit	Compares the 32-bit unsigned binary value in CD and CD+1 (word contents or constant) to the range defined by LL and UL and outputs the results to the Arithmetic Flags in the Auxiliary Area.	Output	Required
SIGNED AREA RANGE COMPARE	ZCPS	117	 **CD**: Compare data (1 word) **LL**: Lower limit of range **UL**: Upper limit of range	Compares the 16-bit signed binary value in CD (word contents or constant) to the range defined by LL and UL and outputs the results to the Arithmetic Flags in the Auxiliary Area.	Output	Required
DOUBLE SIGNED AREA RANGE COMPARE	ZCPSL	118	 **CD**: Compare data (2 words) **LL**: First word of lower limit **UL**: First word of upper limit	Compares the 32-bit signed binary value in CD and CD+1 (word contents or constant) to the range defined by LL and UL and outputs the results to the Arithmetic Flags in the Auxiliary Area.	Output	Required

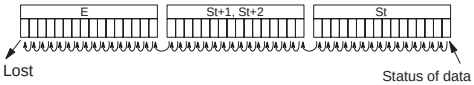
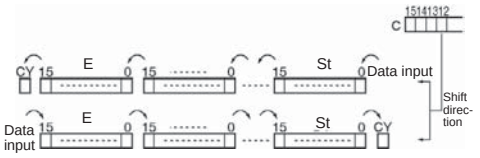
A-1-6 Data Movement Instructions

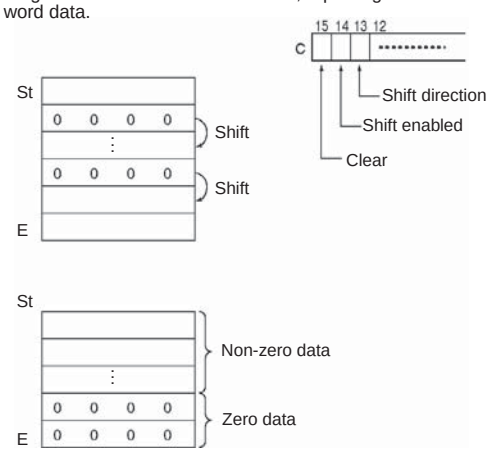
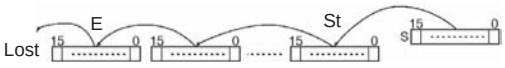
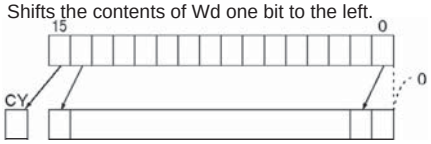
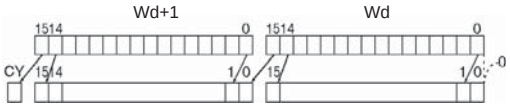
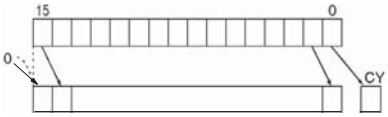
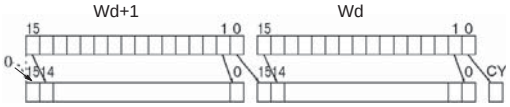
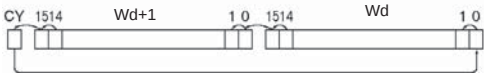
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MOVE	MOV @MOV !MOV !@MOV	021	MOV(021) S D **S**: Source **D**: Destination	Transfers a word of data to the specified word. 	Output	Required
DOUBLE MOVE	MOVL @MOVL	498	MOVL(498) S D **S**: 1st source word **D**: 1st destination word	Transfers two words of data to the specified words. 	Output	Required
MOVE NOT	MVN @MVN	022	MVN(022) S D **S**: Source **D**: Destination	Transfers the complement of a word of data to the specified word. 	Output	Required
DOUBLE MOVE NOT	MVNL @MVNL	499	MVNL(499) S D **S**: 1st source word **D**: 1st destination word	Transfers the complement of two words of data to the specified words. 	Output	Required
MOVE BIT	MOVB @MOVB	082	MOVB(082) S C D **S**: Source word or data **C**: Control word **D**: Destination word	Transfers the specified bit. 	Output	Required
MOVE DIGIT	MOVD @MOVD	083	MOVD(083) S C D **S**: Source word or data **C**: Control word **D**: Destination word	Transfers the specified digit or digits. (Each digit is made up of 4 bits.) 	Output	Required

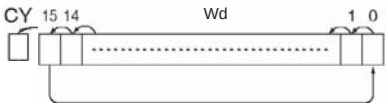
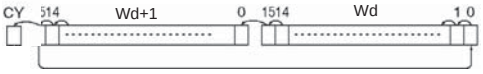
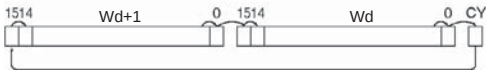
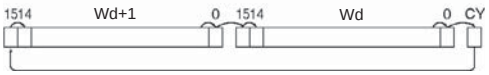
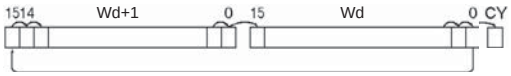
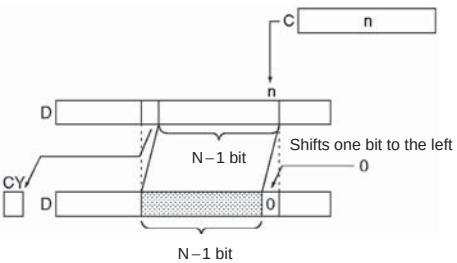
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MULTIPLE BIT TRANSFER	XFRB @XFRB	062	XFRB(062) C S D **C**: Control word **S**: 1st source word **D**: 1st destination word	Transfers the specified number of consecutive bits. 	Output	Required
BLOCK TRANSFER	XFER @XFER	070	XFER(070) N S D **N**: Number of words **S**: 1st source word **D**: 1st destination word	Transfers the specified number of consecutive words. 	Output	Required
BLOCK SET	BSET @BSET	071	BSET(071) S St E **S**: Source word **St**: Starting word **E**: End word	Copies the same word to a range of consecutive words. 	Output	Required
DATA EXCHANGE	XCHG @XCHG	073	XCHG(073) E1 E2 **E1**: 1st exchange word **E2**: Second exchange word	Exchanges the contents of the two specified words. 	Output	Required
DOUBLE DATA EXCHANGE	XCGL @XCGL	562	XCGL(562) E1 E2 **E1**: 1st exchange word **E2**: Second exchange word	Exchanges the contents of a pair of consecutive words with another pair of consecutive words. 	Output	Required
SINGLE WORD DISTRIBUTE	DIST @DIST	080	DIST(080) S Bs Of **S**: Source word **Bs**: Destination base address **Of**: Offset	Transfers the source word to a destination word calculated by adding an offset value to the base address. 	Output	Required

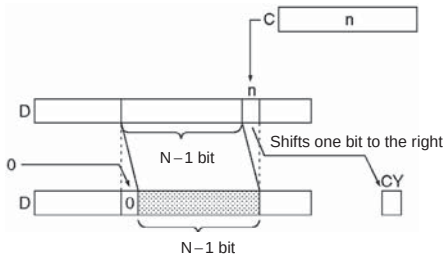
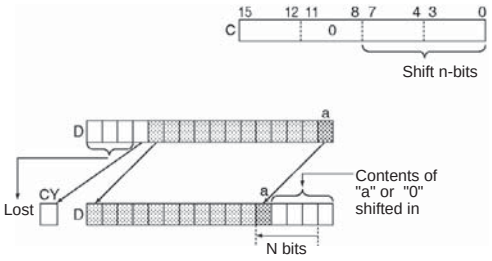
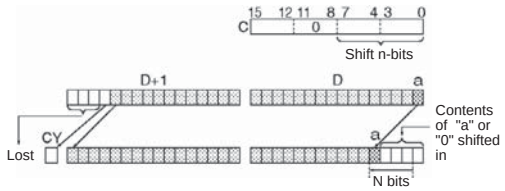
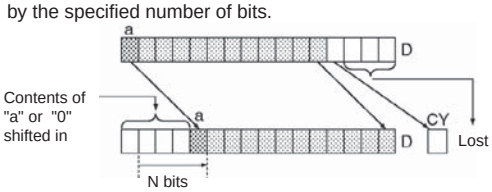
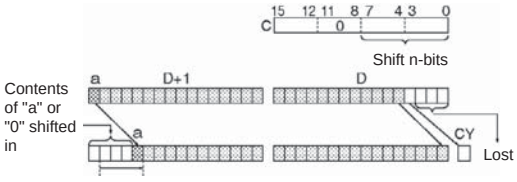
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DATA COLLECT	COLL @COLL	081	COLL(081) Bs Of D **Bs**: Source base address **Of**: Offset **D**: Destination word	Transfers the source word (calculated by adding an offset value to the base address) to the destination word. 	Output	Required
MOVE TO REGISTER	MOVR @MOVR	560	MOVR(560) S D **S**: Source (desired word or bit) **D**: Destination	Sets the internal I/O memory address of the specified word, bit, or timer/counter Completion Flag in the specified Index Register. (Use MOVRW(561) to set the internal I/O memory address of a timer/counter PV in an Index Register.) 	Output	Required
MOVE TIMER/ COUNTER PV TO REGISTER	MOVRW @MOVRW	561	MOVRW(561) S D **S**: Source (desired TC number) **D**: Destination (Index Register)	Sets the internal I/O memory address of the specified timer or counter's PV in the specified Index Register. (Use MOVR(560) to set the internal I/O memory address of a word, bit, or timer/counter Completion Flag in an Index Register.) 	Output	Required

A-1-7 Data Shift Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SHIFT REGISTER	SFT	010	Data input Shift input Reset input SFT(010) St E **St**: Starting word **E**: End word	Operates a shift register. 	Output	Required
REVERSIBLE SHIFT REGISTER	SFTR @SFTR	084	SFTR(084) C St E **C**: Control word **St**: Starting word **E**: End word	Creates a shift register that shifts data to either the right or the left. 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
ASYNCHRO-NOUS SHIFT REGISTER	ASFT @ASFT	017	ASFT(017) C St E **C**: Control word **St**: Starting word **E**: End word	Shifts all non-zero word data within the specified word range either towards St or toward E, replacing 0000Hex word data. 	Output	Required
WORD SHIFT	WSFT @WSFT	016	WSFT(016) S St E **S**: Source word **St**: Starting word **E**: End word	Shifts data between St and E in word units. 	Output	Required
ARITHMETIC SHIFT LEFT	ASL @ASL	025	ASL(025) Wd **Wd**: Word	Shifts the contents of Wd one bit to the left. 	Output	Required
DOUBLE SHIFT LEFT	ASLL @ASLL	570	ASLL(570) Wd **Wd**: Word	Shifts the contents of Wd and Wd +1 one bit to the left. 	Output	Required
ARITHMETIC SHIFT RIGHT	ASR @ASR	026	ASR(026) Wd **Wd**: Word	Shifts the contents of Wd one bit to the right. 	Output	Required
DOUBLE SHIFT RIGHT	ASRL @ASRL	571	ASRL(571) Wd **Wd**: Word	Shifts the contents of Wd and Wd +1 one bit to the right. 	Output	Required
ROTATE LEFT	ROL @ROL	027	ROL(027) Wd **Wd**: Word	Shifts all Wd bits one bit to the left including the Carry Flag (CY). 	Output	Required
DOUBLE ROTATE LEFT	ROLL @ROLL	572	ROLL(572) Wd **Wd**: Word	Shifts all Wd and Wd +1 bits one bit to the left including the Carry Flag (CY). 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
ROTATE LEFT WITHOUT CARRY	RLNC @RLNC	574	RLNC(574) Wd **Wd**: Word	Shifts all Wd bits one bit to the left not including the Carry Flag (CY). 	Output	Required
DOUBLE ROTATE LEFT WITHOUT CARRY	RLNL @RLNL	576	RLNL(576) Wd **Wd**: Word	Shifts all Wd and Wd +1 bits one bit to the left not including the Carry Flag (CY). 	Output	Required
ROTATE RIGHT	ROR @ROR	028	ROR(028) Wd **Wd**: Word	Shifts all Wd bits one bit to the right including the Carry Flag (CY). 	Output	Required
DOUBLE ROTATE RIGHT	RORL @RORL	573	RORL(573) Wd **Wd**: Word	Shifts all Wd and Wd +1 bits one bit to the right including the Carry Flag (CY). 	Output	Required
ROTATE RIGHT WITHOUT CARRY	RRNC @RRNC	575	RRNC(575) Wd **Wd**: Word	Shifts all Wd bits one bit to the right not including the Carry Flag (CY). The contents of the rightmost bit of Wd shifts to the leftmost bit and to the Carry Flag (CY). 	Output	Required
DOUBLE ROTATE RIGHT WITHOUT CARRY	RRNL @RRNL	577	RRNL(577) Wd **Wd**: Word	Shifts all Wd and Wd +1 bits one bit to the right not including the Carry Flag (CY). The contents of the rightmost bit of Wd +1 is shifted to the leftmost bit of Wd, and to the Carry Flag (CY). 	Output	Required
ONE DIGIT SHIFT LEFT	SLD @SLD	074	SLD(074) St E **St**: Starting word **E**: End word	Shifts data by one digit (4 bits) to the left. 	Output	Required
ONE DIGIT SHIFT RIGHT	SRD @SRD	075	SRD(075) St E **St**: Starting word **E**: End word	Shifts data by one digit (4 bits) to the right. 	Output	Required
SHIFT N-BIT DATA LEFT	NSFL @NSFL	578	NSFL(578) D C N **D**: Beginning word for shift **C**: Beginning bit **N**: Shift data length	Shifts the specified number of bits to the left. 	Output	Required

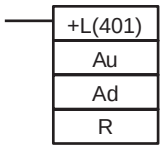
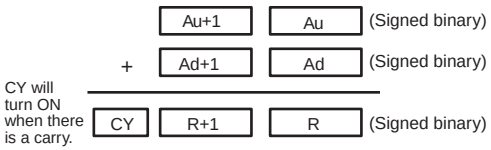
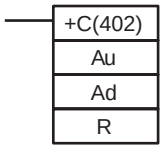
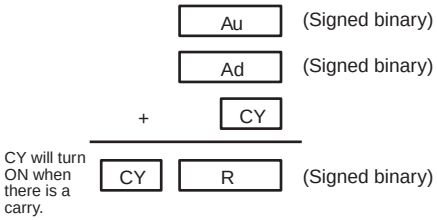
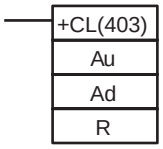
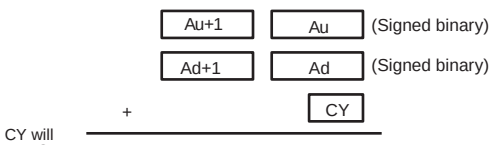
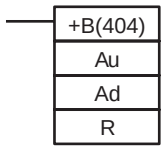
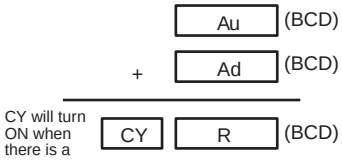
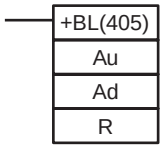
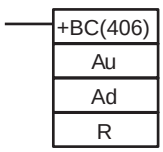
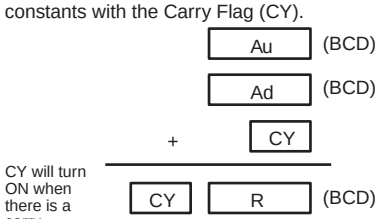
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SHIFT N-BIT DATA RIGHT	NSFR @NSFR	579	NSFR(579) D C N **D**: Beginning word for shift **C**: Beginning bit **N**: Shift data length	Shifts the specified number of bits to the right. 	Output	Required
SHIFT N-BITS LEFT	NASL @NASL	580	NASL(580) D C **D**: Shift word **C**: Control word	Shifts the specified 16 bits of word data to the left by the specified number of bits. 	Output	Required
DOUBLE SHIFT N-BITS LEFT	NSLL @NSLL	582	NSLL(582) D C **D**: Shift word **C**: Control word	Shifts the specified 32 bits of word data to the left by the specified number of bits. 	Output	Required
SHIFT N-BITS RIGHT	NASR @NASR	581	NASR(581) D C **D**: Shift word **C**: Control word	Shifts the specified 16 bits of word data to the right by the specified number of bits. 	Output	Required
DOUBLE SHIFT N-BITS RIGHT	NSRL @NSRL	583	NSRL(583) D C **D**: Shift word **C**: Control word	Shifts the specified 32 bits of word data to the right by the specified number of bits. 	Output	Required

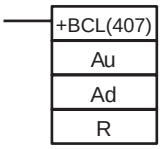
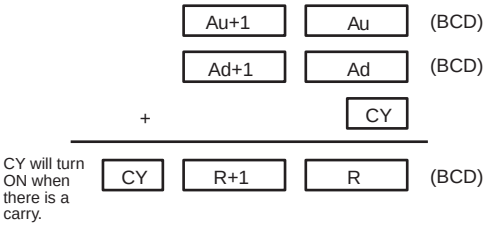
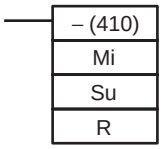
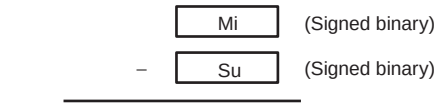
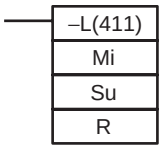
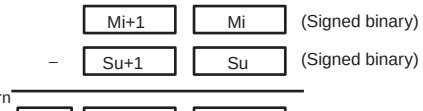
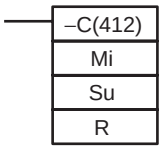
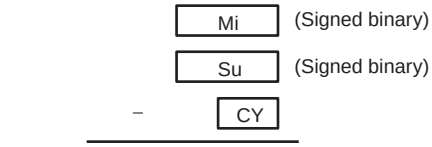
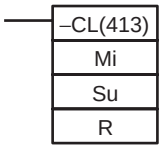
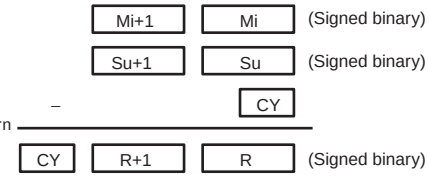
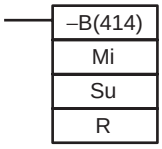
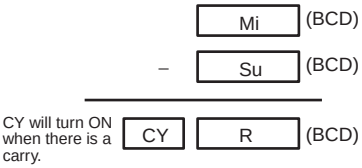
A-1-8 Increment/Decrement Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
INCREMENT BINARY	++ @++	590	++(590) Wd **Wd**: Word	Increments the 4-digit hexadecimal content of the specified word by 1. Wd +1 → Wd	Output	Required
DOUBLE INCREMENT BINARY	++L @++L	591	++L(591) Wd **Wd**: Word	Increments the 8-digit hexadecimal content of the specified words by 1. Wd+1 Wd +1 → Wd+1 Wd	Output	Required
DECREMENT BINARY	– – @– –	592	– – (592) Wd **Wd**: Word	Decrements the 4-digit hexadecimal content of the specified word by 1. Wd –1 → Wd	Output	Required
DOUBLE DECREMENT BINARY	– –L @– –L	593	– – L(593) Wd **Wd**: 1st word	Decrements the 8-digit hexadecimal content of the specified words by 1. Wd+1 Wd –1 → Wd+1 Wd	Output	Required
INCREMENT BCD	++B @++B	594	++B(594) Wd **Wd**: Word	Increments the 4-digit BCD content of the specified word by 1. Wd +1 → Wd	Output	Required
DOUBLE INCREMENT BCD	++BL @++BL	595	++BL(595) Wd **Wd**: 1st word	Increments the 8-digit BCD content of the specified words by 1. Wd+1 Wd +1 → Wd+1 Wd	Output	Required
DECREMENT BCD	– –B @– –B	596	– – B(596) Wd **Wd**: Word	Decrements the 4-digit BCD content of the specified word by 1. Wd –1 → Wd	Output	Required
DOUBLE DECREMENT BCD	– –BL @– –BL	597	– – BL(597) Wd **Wd**: 1st word	Decrements the 8-digit BCD content of the specified words by 1. Wd+1 Wd –1 → Wd+1 Wd	Output	Required

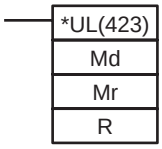
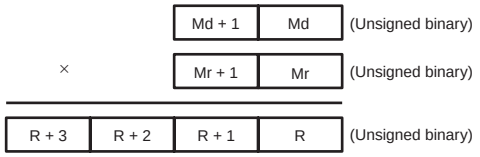
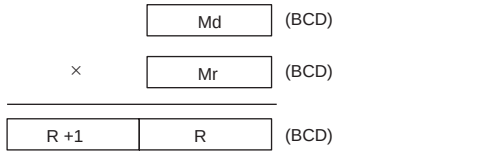
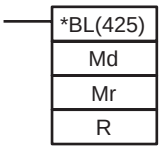
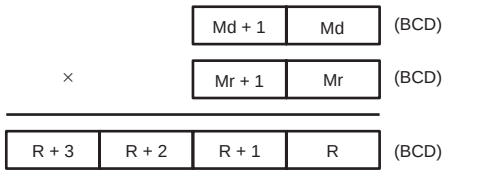
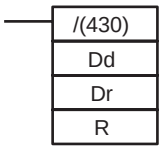
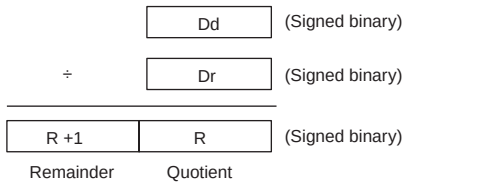
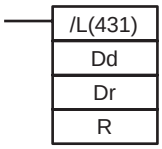
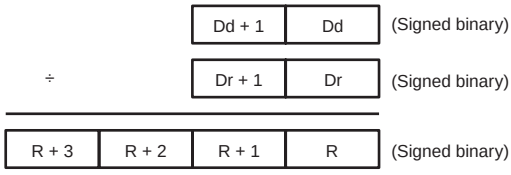
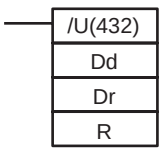
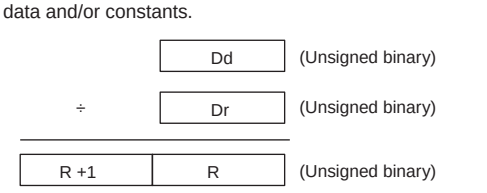
A-1-9 Symbol Math Instructions

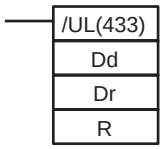
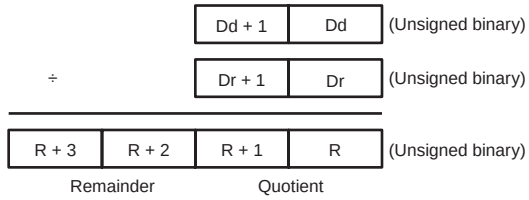
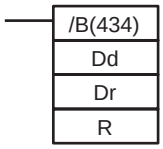
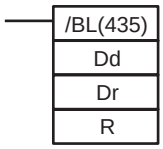
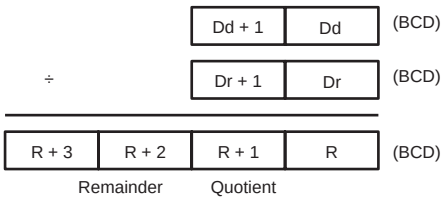
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SIGNED BINARY ADD WITHOUT CARRY	+ @+	400	+(400) Au Ad R **Au**: Augend word **Ad**: Addend word **R**: 1st result word	Adds 4-digit (single-word) hexadecimal data and/or constants. Au (Signed binary) + Ad (Signed binary) CY will turn ON when there is a carry. CY R (Signed binary)	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE SIGNED BINARY ADD WITHOUT CARRY	+L @+L	401	 **Au**: 1st augend word **Ad**: 1st addend word **R**: 1st result word	Adds 8-digit (double-word) hexadecimal data and/or constants. 	Output	Required
SIGNED BINARY ADD WITH CARRY	+C @+C	402	 **Au**: Augend word **Ad**: Addend word **R**: Result word	Adds 4-digit (single-word) hexadecimal data and/or constants with the Carry Flag (CY). 	Output	Required
DOUBLE SIGNED BINARY ADD WITH CARRY	+CL @+CL	403	 **Au**: 1st augend word **Ad**: 1st addend word **R**: 1st result word	Adds 8-digit (double-word) hexadecimal data and/or constants with the Carry Flag (CY). 	Output	Required
BCD ADD WITHOUT CARRY	+B @+B	404	 **Au**: Augend word **Ad**: Addend word **R**: Result word	Adds 4-digit (single-word) BCD data and/or constants. 	Output	Required
DOUBLE BCD ADD WITH-OUT CARRY	+BL @+BL	405	 **Au**: 1st augend word **Ad**: 1st addend word **R**: 1st result word	Adds 8-digit (double-word) BCD data and/or constants. 	Output	Required
BCD ADD WITH CARRY	+BC @+BC	406	 **Au**: Augend word **Ad**: Addend word **R**: Result word	Adds 4-digit (single-word) BCD data and/or constants with the Carry Flag (CY). 	Output	Required

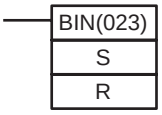
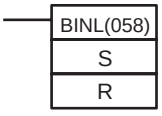
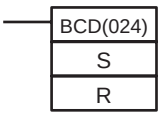
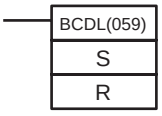
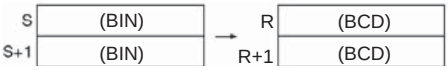
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE BCD ADD WITH CARRY	+BCL @+BCL	407	 **Au**: 1st augend word **Ad**: 1st addend word **R**: 1st result word	Adds 8-digit (double-word) BCD data and/or constants with the Carry Flag (CY). 	Output	Required
SIGNED BINARY SUB-TRACT WITH-OUT CARRY	– @–	410	 **Mi**: Minuend word **Su**: Subtrahend word	Subtracts 4-digit (single-word) hexadecimal data and/or constants. 	Output	Required
DOUBLE SIGNED BINARY SUB-TRACT WITH-OUT CARRY	– L @– L	411	 **Mi**: Minuend word **Su**: Subtrahend word **R**: Result word	Subtracts 8-digit (double-word) hexadecimal data and/or constants. 	Output	Required
SIGNED BINARY SUB-TRACT WITH CARRY	– C @– C	412	 **Mi**: Minuend word **Su**: Subtrahend word **R**: Result word	Subtracts 4-digit (single-word) hexadecimal data and/or constants with the Carry Flag (CY). 	Output	Required
DOUBLE SIGNED BINARY WITH CARRY	– CL @–- CL	413	 **Mi**: Minuend word **Su**: Subtrahend word **R**: Result word	Subtracts 8-digit (double-word) hexadecimal data and/or constants with the Carry Flag (CY). 	Output	Required
BCD SUB-TRACT WITH-OUT CARRY	– B @– B	414	 **Mi**: Minuend word **Su**: Subtrahend word **R**: Result word	Subtracts 4-digit (single-word) BCD data and/or constants. 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE BCD SUBTRACT WITHOUT CARRY	– BL @– BL	415	–BL(415) Mi Su R **Mi**: 1st minuend word **Su**: 1st subtrahend word **R**: 1st result word	Subtracts 8-digit (double-word) BCD data and/or constants. Mi +1 Mi (BCD) – Su+1 Su (BCD) CY will turn ON when there is a borrow. CY R+1 R (BCD)	Output	Required
BCD SUBTRACT WITH CARRY	– BC @– BC	416	–BC(416) Mi Su R **Mi**: Minuend word **Su**: Subtrahend word **R**: Result word	Subtracts 4-digit (single-word) BCD data and/or constants with the Carry Flag (CY). Mi (BCD) Su (BCD) – CY CY will turn ON when there is a borrow. CY R (BCD)	Output	Required
DOUBLE BCD SUBTRACT WITH CARRY	– BCL @– BCL	417	–BCL(417) Mi Su R **Mi**: 1st minuend word **Su**: 1st subtrahend word **R**: 1st result word	Subtracts 8-digit (double-word) BCD data and/or constants with the Carry Flag (CY). Mi +1 Mi (BCD) Su+1 Su (BCD) – CY CY will turn ON when there is a borrow. CY R+1 R (BCD)	Output	Required
SIGNED BINARY MULTIPLY	* @*	420	*(420) Md Mr R **Md**: Multiplicand word **Mr**: Multiplier word **R**: Result word	Multiplies 4-digit signed hexadecimal data and/or constants. Md (Signed binary) × Mr (Signed binary) R +1 | R (Signed binary)	Output	Required
DOUBLE SIGNED BINARY MULTIPLY	*L @*L	421	*L(421) Md Mr R **Md**: 1st multiplicand word **Mr**: 1st multiplier word **R**: 1st result word	Multiplies 8-digit signed hexadecimal data and/or constants. Md + 1 | Md (Signed binary) × Mr + 1 | Mr (Signed binary) R + 3 | R + 2 | R + 1 | R (Signed binary)	Output	Required
UNSIGNED BINARY MULTIPLY	*U @*U	422	*U(422) Md Mr R **Md**: Multiplicand word **Mr**: Multiplier word **R**: Result word	Multiplies 4-digit unsigned hexadecimal data and/or constants. Md (Unsigned binary) × Mr (Unsigned binary) R +1 | R (Unsigned binary)	Output	Required

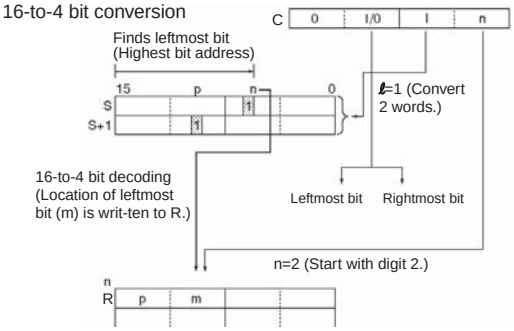
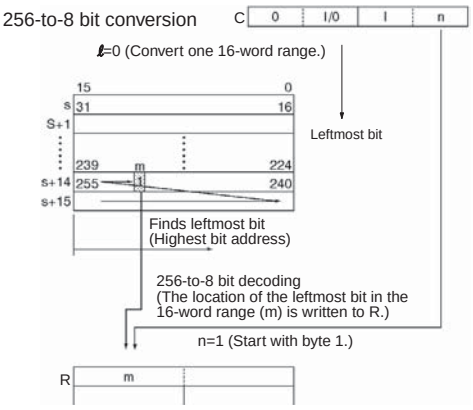
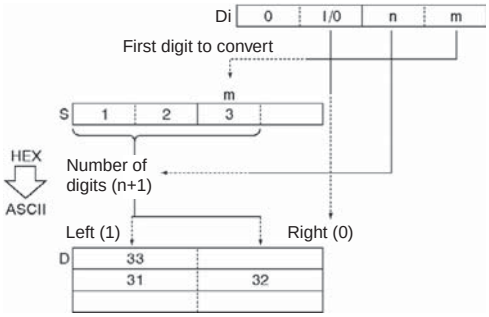
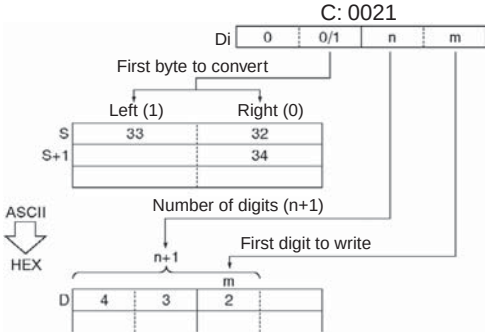
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE UNSIGNED BINARY MULTIPLY	*UL @*UL	423	 **Md**: 1st multiplicand word **Mr**: 1st multiplier word **R**: 1st result word	Multiplies 8-digit unsigned hexadecimal data and/or constants. 	Output	Required
BCD MULTIPLY	*B @*B	424	 **Md**: Multiplicand word **Mr**: Multiplier word **R**: Result word	Multiplies 4-digit (single-word) BCD data and/or constants. 	Output	Required
DOUBLE BCD MULTIPLY	*BL @*BL	425	 **Md**: 1st multiplicand word **Mr**: 1st multiplier word **R**: 1st result word	Multiplies 8-digit (double-word) BCD data and/or constants. 	Output	Required
SIGNED BINARY DIVIDE	/ @/	430	 **Dd**: Dividend word **Dr**: Divisor word **R**: Result word	Divides 4-digit (single-word) signed hexadecimal data and/or constants. 	Output	Required
DOUBLE SIGNED BINARY DIVIDE	/L @/L	431	 **Dd**: 1st dividend word **Dr**: 1st divisor word **R**: 1st result word	Divides 8-digit (double-word) signed hexadecimal data and/or constants. 	Output	Required
UNSIGNED BINARY DIVIDE	/U @/U	432	 **Dd**: Dividend word **Dr**: Divisor word **R**: Result word	Divides 4-digit (single-word) unsigned hexadecimal data and/or constants. 	Output	Required

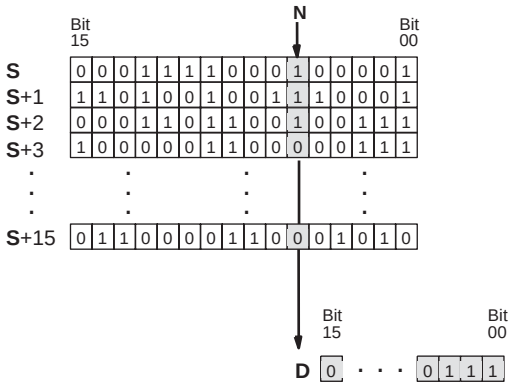
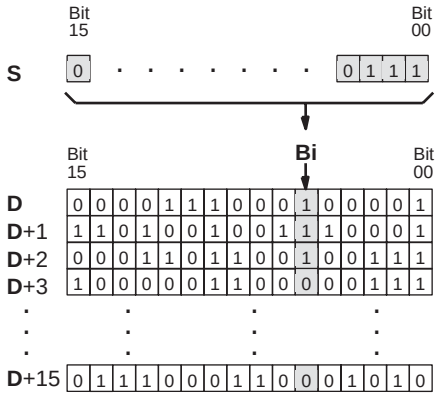
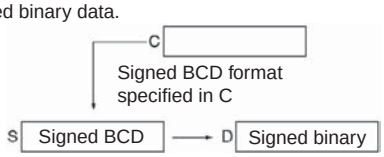
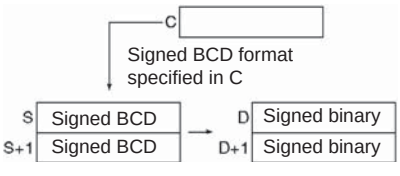
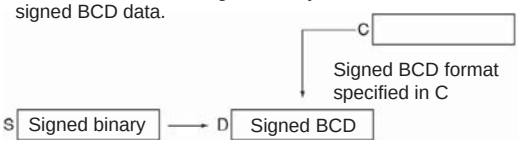
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE UNSIGNED BINARY DIVIDE	/UL @/UL	433	 **Dd**: 1st dividend word **Dr**: 1st divisor word **R**: 1st result word	Divides 8-digit (double-word) unsigned hexadecimal data and/or constants. 	Output	Required
BCD DIVIDE	/B @/B	434	 **Dd**: Dividend word **Dr**: Divisor word **R**: Result word	Divides 4-digit (single-word) BCD data and/or constants. 	Output	Required
DOUBLE BCD DIVIDE	/BL @/BL	435	 **Dd**: 1st dividend word **Dr**: 1st divisor word **R**: 1st result word	Divides 8-digit (double-word) BCD data and/or constants. 	Output	Required

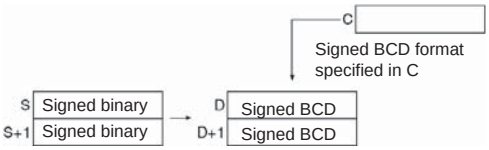
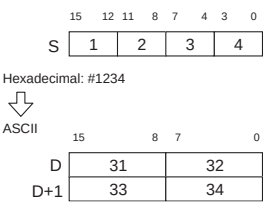
A-1-10 Conversion Instructions

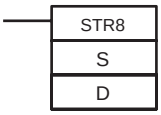
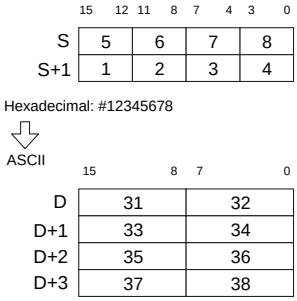
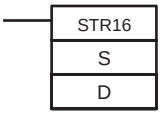
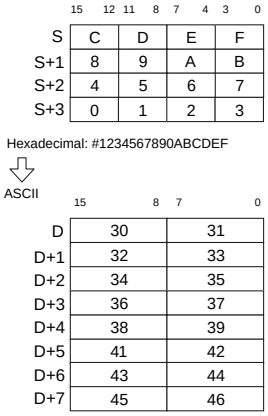
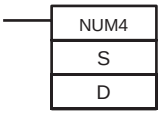
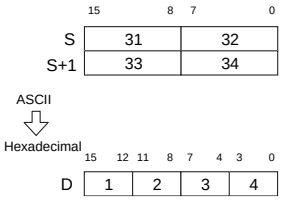
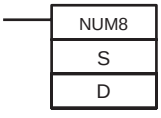
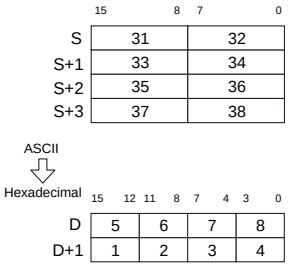
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
BCD-TO-BINARY	BIN @BIN	023	 **S**: Source word **R**: Result word	Converts BCD data to binary data. 	Output	Required
DOUBLE BCD-TO-DOUBLE BINARY	BINL @BINL	058	 **S**: 1st source word **R**: 1st result word	Converts 8-digit BCD data to 8-digit hexadecimal (32-bit binary) data. 	Output	Required
BINARY-TO-BCD	BCD @BCD	024	 **S**: Source word **R**: Result word	Converts a word of binary data to a word of BCD data. 	Output	Required
DOUBLE BINARY-TO-DOUBLE BCD	BCDL @BCDL	059	 **S**: 1st source word **R**: 1st result word	Converts 8-digit hexadecimal (32-bit binary) data to 8-digit BCD data. 	Output	Required

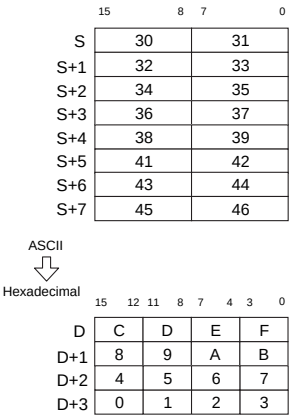
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
2'S COMPLEMENT	NEG @NEG	160	NEG(160) S R **S**: Source word **R**: Result word	Calculates the 2's complement of a word of hexadecimal data. 2's complement (Complement + 1) $\overline{(S)}$ → (R)	Output	Required
DOUBLE 2'S COMPLEMENT	NEGL @NEGL	161	NEGL(161) S R **S**: 1st source word **R**: 1st result word	Calculates the 2's complement of two words of hexadecimal data. 2's complement (Complement + 1) $\overline{(S+1, S)}$ → (R+1, R)	Output	Required
16-BIT TO 32-BIT SIGNED BINARY	SIGN @SIGN	600	SIGN(600) S R **S**: Source word **R**: 1st result word	Expands a 16-bit signed binary value to its 32-bit equivalent. MSB = 1: FFFF Hex MSB = 0: 0000 Hex D+1 D D = Contents of S	Output	Required
DATA DECODER	MLPX @MLPX	076	MLPX(076) S C R **S**: Source word **C**: Control word **R**: 1st result word	Reads the numerical value in the specified digit (or byte) in the source word, turns ON the corresponding bit in the result word (or 16-word range), and turns OFF all other bits in the result word (or 16-word range). 4-to-16 bit conversion ℓ=1 (Convert 2 digits.) n=2 (Start with second digit.) 4-to-16 bit decoding (Bit m of R is turned ON.) 8-to-256 bit conversion ℓ=1 (Convert 2 bytes.) n=1 (Start with first byte.) 8-to-256 bit decoding (Bit m of R to R+15 is turned ON.) Two 16-word ranges are used when ℓ specifies 2 bytes.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DATA ENCODER	DMPX @DMPX	077	DMPX(077) S R C **S**: 1st source word **R**: Result word **C**: Control word	Finds the location of the first or last ON bit within the source word (or 16-word range), and writes that value to the specified digit (or byte) in the result word. 16-to-4 bit conversion  256-to-8 bit conversion 	Output	Required
ASCII CON-VERT	ASC @ASC	086	ASC(086) S Di D **S**: Source word **Di**: Digit designator **D**: 1st destination word	Converts 4-bit hexadecimal digits in the source word into their 8-bit ASCII equivalents. 	Output	Required
ASCII TO HEX	HEX @HEX	162	HEX(162) S Di D **S**: 1st source word **Di**: Digit designator **D**: Destination word	Converts up to 4 bytes of ASCII data in the source word to their hexadecimal equivalents and writes these digits in the specified destination word. 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
COLUMN TO LINE	LINE @LINE	063	LINE(063) S N D **S**: 1st source word **N**: Bit number **D**: Destination word	Converts a column of bits from a 16-word range (the same bit number in 16 consecutive words) to the 16 bits of the destination word. 	Output	Required
LINE TO COLUMN	COLM @COLM	064	COLM(064) S D N **S**: Source word **D**: 1st destination word **N**: Bit number	Converts the 16 bits of the source word to a column of bits in a 16-word range of destination words (the same bit number in 16 consecutive words). 	Output	Required
SIGNED BCD-TO-BINARY	BINS @BINS	470	BINS(470) C S D **C**: Control word **S**: Source word **D**: Destination word	Converts one word of signed BCD data to one word of signed binary data. 	Output	Required
DOUBLE SIGNED BCD-TO-BINARY	BISL @BISL	472	BISL(472) C S D **C**: Control word **S**: 1st source word **D**: 1st destination word	Converts double signed BCD data to double signed binary data. 	Output	Required
SIGNED BINARY-TO-BCD	BCDS @BCDS	471	BCDS(471) C S D **C**: Control word **S**: Source word **D**: Destination word	Converts one word of signed binary data to one word of signed BCD data. 	Output	Required

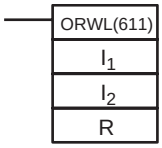
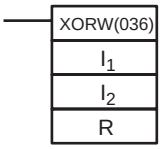
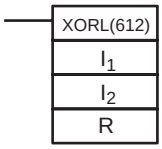
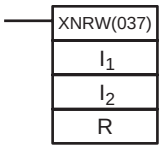
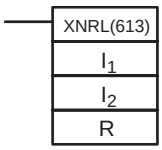
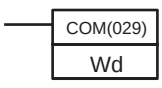
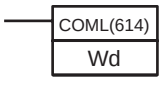
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE SIGNED BINARY-TO-BCD	BDSL @BDSL	473	BDSL(473) C S D **C**: Control word **S**: 1st source word **D**: 1st destination word	Converts double signed binary data to double signed BCD data. 	Output	Required
GRAY CODE CONVERSION	GRY	474	GRY (474) C S D **C**: Control word **S**: Source word **D**: 1st destination word	Converts the Gray code data in the specified word to binary, BCD, or angle (?) data at the specified resolution.	Output	Required
GRAY CODE TOBINARY CONVERT	GRAY _BIN @GRAY_ BIN	478	GRAY_BIN(478) S D **S**: Source (gray code) **D**: Destination (binary data)	Converts the word of gray code specified by S to one word of binary data, and outputs it to D.	Output	Required
DOUBLE GRAYCODE TO BINARY-CONVERT	GRAY _BINL @GRAY_ BINL	479	GRAY_BINL(479) S D **S**: 1st source word (gray code) **D**: 1st destination word (binary)	Converts the two words of gray code specified by S to two words of binary data, and outputs them to D.	Output	Required
BINARY TO GRAYCODE CONVERT	BIN _GRAY @BIN_ GRAY	480	BIN_GRAY(480) S D **S**: Source word (binary data) **D**: Destination word (gray code)	Converts the word of binary data specified by S to one word of gray code, and outputs it to D.	Output	Required
DOUBLE BINA-RYTO GRAY CODECON-VERT	BIN _GRAYL @BIN_ GRAYL	481	BIN_GRAYL(481) S D **S**: 1st source word (binary data) **D**: 1st destination word (gray code)	Converts the two words of binary data specified by S to two words of gray code, and outputs them to D.	Output	Required
FOUR-DIGIT NUMBER TO ASCII	STR4 @STR4	601	STR4 S D **S**: Numeric **D**: ASCII text	Converts a 4-digit hexadecimal number (#0000 to #FFFF) to ASCII data (4 characters). 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
EIGHT-DIGIT NUMBER TO ASCII	STR8 @STR8	602	 **S**: Numeric **D**: ASCII text	Converts an 8-digit hexadecimal number (#0000 0000 to #FFFF FFFF) to ASCII data (8 characters). 	Output	Required
SIXTEEN-DIGIT NUMBER TO ASCII	STR16 @STR16	603	 **S**: Numeric **D**: ASCII text	Converts a 16-digit hexadecimal number (#0000 0000 0000 0000 to #FFFF FFFF FFFF FFFF) to ASCII data (16 characters). 	Output	Required
ASCII TO FOUR-DIGIT NUMBER	NUM4 @NUM4	604	 **S**: ASCII text **D**: Numeric	Converts 4 characters of ASCII data to a 4-digit hexadecimal number. 	Output	Required
ASCII TO EIGHT-DIGIT NUMBER	NUM8 @NUM8	605	 **S**: ASCII text **D**: Numeric	Converts 8 characters of ASCII data to an 8-digit hexadecimal number. 	Output	Required

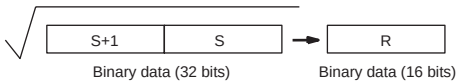
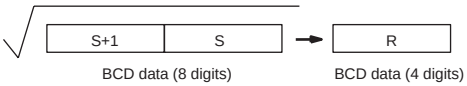
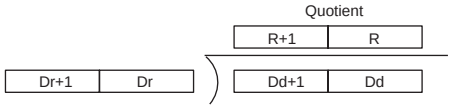
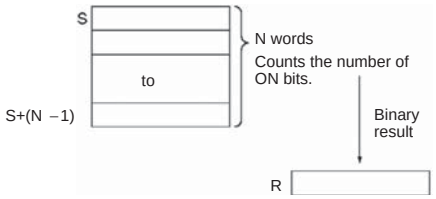
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
ASCII TO SIX-TEEN-DIGIT-NUMBER	NUM16 @NUM16	606	STR16(606) S D **S**: ASCII text **D**: Numeric	Converts 16 characters of ASCII data to a 16-digit hexadecimal number. 	Output	Required

A-1-11 Logic Instructions

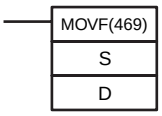
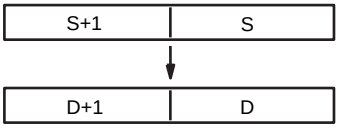
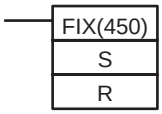
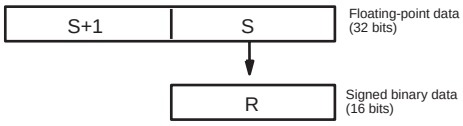
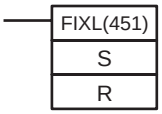
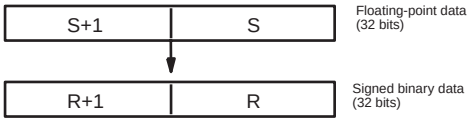
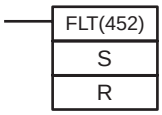
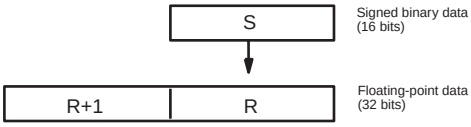
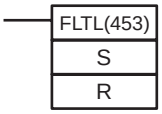
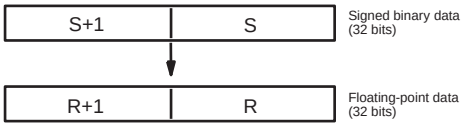
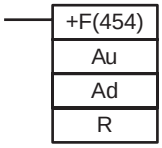
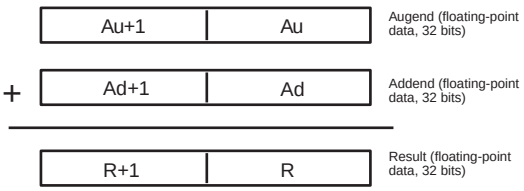
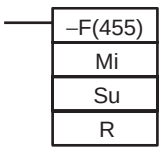
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LOGICAL AND	ANDW @ANDW	034	ANDW(034) I_1 I_2 R **I1**: Input 1 **I2**: Input 2 **R**: Result word	Takes the logical AND of corresponding bits in single words of word data and/or constants. **$I_1 \cdot I_2 \rightarrow R$** I_1 I_2 R 1 1 1 1 0 0 0 1 0 0 0 0	Output	Required
DOUBLE LOGICAL AND	ANDL @ANDL	610	ANDL(610) I_1 I_2 R **I1**: Input 1 **I2**: Input 2 **R**: Result word	Takes the logical AND of corresponding bits in double words of word data and/or constants. **$(I_1, I_1+1) \cdot (I_2, I_2+1) \rightarrow (R, R+1)$** I_1, I_1+1 I_2, I_2+1 R, R+1 1 1 1 1 0 0 0 1 0 0 0 0	Output	Required
LOGICAL OR	ORW @ORW	035	ORW(035) I_1 I_2 R **I1**: Input 1 **I2**: Input 2 **R**: Result word	Takes the logical OR of corresponding bits in single words of word data and/or constants. **$I_1 + I_2 \rightarrow R$** I_1 I_2 R 1 1 1 1 0 1 0 1 1 0 0 0	Output	Required

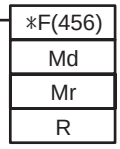
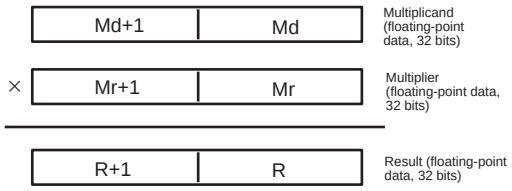
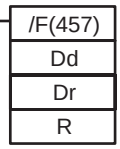
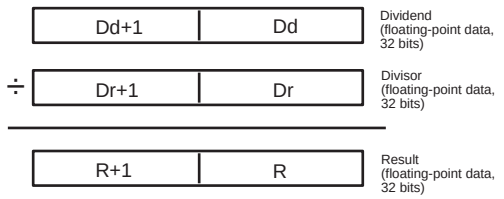
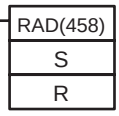
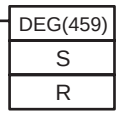
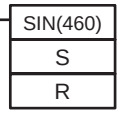
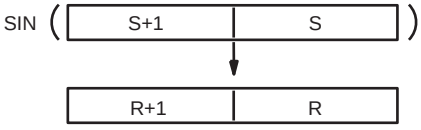
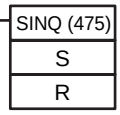
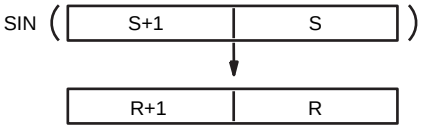
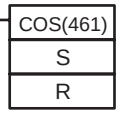
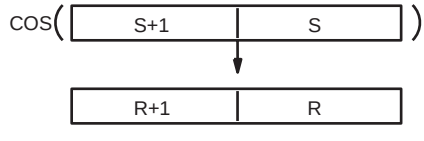
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE LOGICAL OR	ORWL @ORWL	611	 **I_1**: Input 1 **I_2**: Input 2 **R**: Result word	Takes the logical OR of corresponding bits in double words of word data and/or constants. **$(I_1, I_1+1) + (I_2, I_2+1) \rightarrow (R, R+1)$** I_1, I_1+1 / I_2, I_2+1 / R, R+1: 1 1 1; 1 0 1; 0 1 1; 0 0 0	Output	Required
EXCLUSIVE OR	XORW @XORW	036	 **I_1**: Input 1 **I_2**: Input 2 **R**: Result word	Takes the logical exclusive OR of corresponding bits in single words of word data and/or constants. **$I_1.I_2 + I_1.I_2 \rightarrow R$** I_1 / I_2 / R: 1 1 0; 1 0 1; 0 1 1; 0 0 0	Output	Required
DOUBLE EXCLUSIVE OR	XORL @XORL	612	 **I_1**: Input 1 **I_2**: Input 2 **R**: Result word	Takes the logical exclusive OR of corresponding bits in double words of word data and/or constants. **$(I_1, I_1+1).\ (\overline{I_2, I_2+1}) + (\overline{I_1, I_1+1}).\ (I_2, I_2+1) \rightarrow (R, R+1)$** I_1, I_1+1 / I_2, I_2+1 / R, R+1: 1 1 0; 1 0 1; 0 1 1; 0 0 0	Output	Required
EXCLUSIVE NOR	XNRW @XNRW	037	 **I_1**: Input 1 **I_2**: Input 2 **R**: Result word	Takes the logical exclusive NOR of corresponding single words of word data and/or constants. **$I_1.I_2 + I_1.I_2 \rightarrow R$** I_1 / I_2 / R: 1 1 1; 1 0 0; 0 1 0; 0 0 1	Output	Required
DOUBLE EXCLUSIVE NOR	XNRL @XNRL	613	 **I_1**: Input 1 **I_2**: Input 2 **R**: 1st result word	Takes the logical exclusive NOR of corresponding bits in double words of word data and/or constants. **$(I_1, I_1+1).\ (I_2, I_2+1) + (\overline{I_1, I_1+1}).\ (\overline{I_2, I_2+1}) \rightarrow (R, R+1)$** I_1, I_1+1 / I_2, I_2+1 / R, R+1: 1 1 1; 1 0 0; 0 1 0; 0 0 1	Output	Required
COMPLEMENT	COM @COM	029	 **Wd**: Word	Turns OFF all ON bits and turns ON all OFF bits in Wd. $\overline{Wd} \rightarrow Wd$: $1 \rightarrow 0$ and $0 \rightarrow 1$	Output	Required
DOUBLE COM-PLEMENT	COML @COML	614	 **Wd**: Word	Turns OFF all ON bits and turns ON all OFF bits in Wd and Wd+1. $\overline{(Wd+1, Wd)} \rightarrow (Wd+1, Wd)$	Output	Required

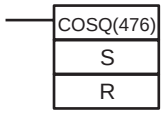
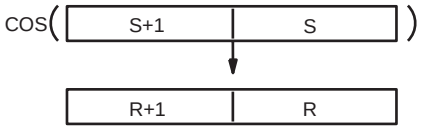
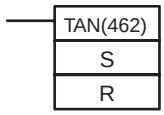
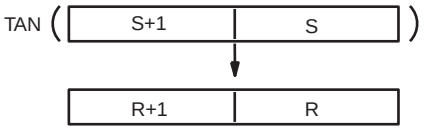
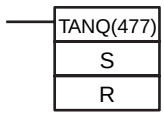
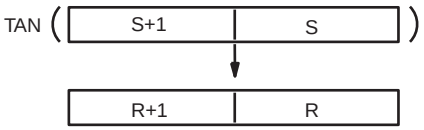
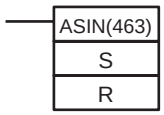
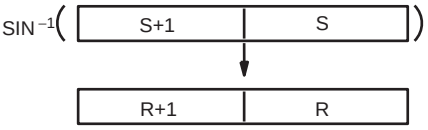
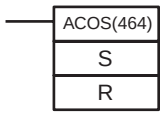
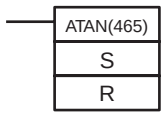
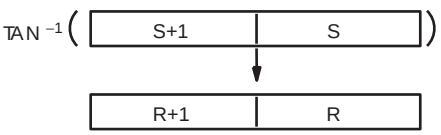
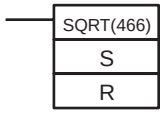
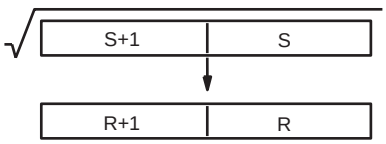
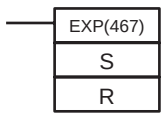
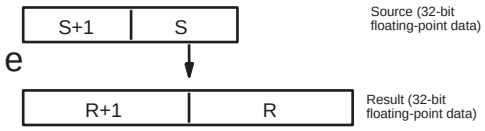
A-1-12 Special Math Instructions

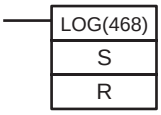
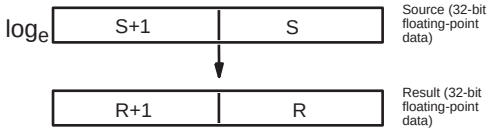
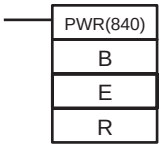
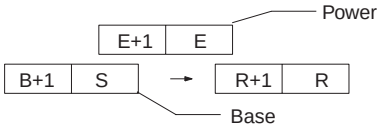
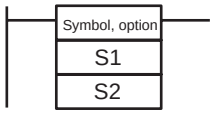
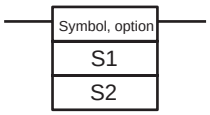
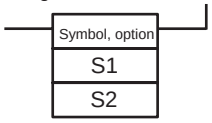
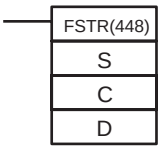
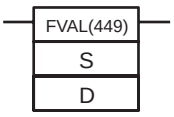
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
BINARY ROOT	ROTB @ROTB	620	ROTB(620) S R **S**: 1st source word **R**: Result word	Computes the square root of the 32-bit binary content of the specified words and outputs the integer portion of the result to the specified result word. 	Output	Required
BCD SQUARE ROOT	ROOT @ROOT	072	ROOT(072) S R **S**: 1st source word **R**: Result word	Computes the square root of an 8-digit BCD number and outputs the integer portion of the result to the specified result word. 	Output	Required
ARITHMETIC PROCESS	APR @APR	069	APR(069) C S R **C**: Control word **S**: Source data **R**: Result word	Calculates SIN, COS, or linear extrapolation. SIN or COS calculation: Calculates the SIN or COS from angle data (0° to 90°) and outputs the result in BCD to four places below the decimal. Linear extrapolation: Calculates and outputs a linear extrapolation in binary from the specified input data.	Output	Required
FLOATING POINT DIVIDE	FDIV @FDIV	079	FDIV(079) Dd Dr R **Dd**: 1st dividend word **Dr**: 1st divisor word **R**: 1st result word	Divides one 7-digit floating-point number by another. The floating-point numbers are expressed in scientific notation (7-digit mantissa and 1-digit exponent). 	Output	Required
BIT COUNTER	BCNT @BCNT	067	BCNT(067) N S R **N**: Number of words **S**: 1st source word **R**: Result word	Counts the total number of ON bits in the specified word(s). 	Output	Required

A-1-13 Floating-point Math Instructions

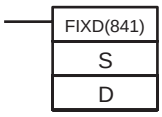
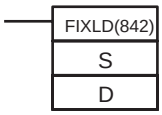
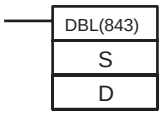
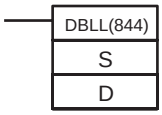
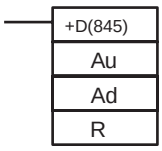
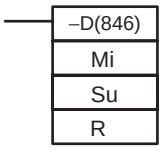
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MOVE FLOATING-POINT (SINGLE)	MOVF	469	 **S**: 1st source word **D**: 1st destination word	Transfers the specified 32-bit floating-point number to the specified destination words. 	Output	Required
FLOATING TO 16-BIT	FIX @FIX	450	 **S**: 1st source word **R**: Result word	Converts a 32-bit floating-point value to 16-bit signed binary data and places the result in the specified result word. 	Output	Required
FLOATING TO 32-BIT	FIXL @FIXL	451	 **S**: 1st source word **R**: 1st result word	Converts a 32-bit floating-point value to 32-bit signed binary data and places the result in the specified result words. 	Output	Required
16-BIT TO FLOATING	FLT @FLT	452	 **S**: Source word **R**: 1st result word	Converts a 16-bit signed binary value to 32-bit floating-point data and places the result in the specified result words. 	Output	Required
32-BIT TO FLOATING	FLTL @FLTL	453	 **S**: 1st source word **R**: 1st result word	Converts a 32-bit signed binary value to 32-bit floating-point data and places the result in the specified result words. 	Output	Required
FLOATING-POINT ADD	+F @+F	454	 **Au**: 1st augend word **AD**: 1st addend word **R**: 1st result word	Adds two 32-bit floating-point numbers and places the result in the specified result words. 	Output	Required
FLOATING-POINT SUBTRACT	–F @–F	455	 **Mi**: 1st Minuend word **Su**: 1st Subtrahend word **R**: 1st result word	Subtracts one 32-bit floating-point number from another and places the result in the specified result words. 	Output	Required

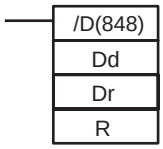
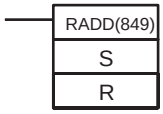
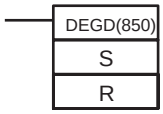
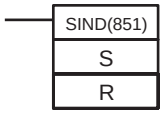
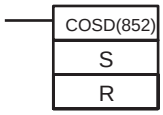
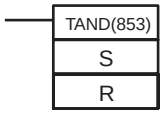
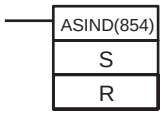
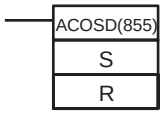
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FLOATING-POINT MULTIPLY	*F @*F	456	 **Md**: 1st Multiplicand word **Mr**: 1st Multiplier word **R**: 1st result word	Multiplies two 32-bit floating-point numbers and places the result in the specified result words. 	Output	Required
FLOATING-POINT DIVIDE	/F @/F	457	 **Dd**: 1st Dividend word **Dr**: 1st Divisor word **R**: 1st result word	Divides one 32-bit floating-point number by another and places the result in the specified result words. 	Output	Required
DEGREES TO RADIANS	RAD @RAD	458	 **S**: 1st source word **R**: 1st result word	Converts a 32-bit floating-point number from degrees to radians and places the result in the specified result words. 	Output	Required
RADIANS TO DEGREES	DEG @DEG	459	 **S**: 1st source word **R**: 1st result word	Converts a 32-bit floating-point number from radians to degrees and places the result in the specified result words. 	Output	Required
SINE	SIN @SIN	460	 **S**: 1st source word **R**: 1st result word	Calculates the sine of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required
HIGH-SPEED SINE	SINQ @SINQ	475	 **S**: 1st source word **R**: 1st result word	Calculates the sine of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required
COSINE	COS @COS	461	 **S**: 1st source word **R**: 1st result word	Calculates the cosine of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required

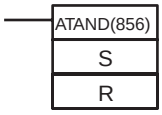
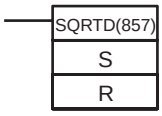
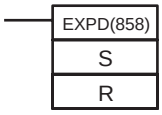
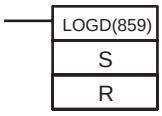
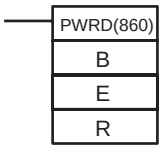
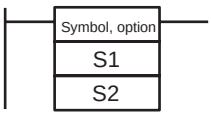
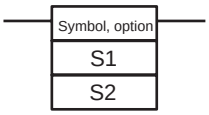
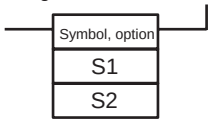
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
HIGH-SPEED COSINE	COSQ @COSQ	476	 **S**: 1st source word **R**: 1st result word	Calculates the cosine of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required
TANGENT	TAN @TAN	462	 **S**: 1st source word **R**: 1st result word	Calculates the tangent of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required
HIGH-SPEED TANGENT	TANQ @TANQ	477	 **S**: 1st source word **R**: 1st result word	Calculates the tangent of a 32-bit floating-point number (in radians) and places the result in the specified result words. 	Output	Required
ARC SINE	ASIN @ASIN	463	 **S**: 1st source word **R**: 1st result word	Calculates the arc sine of a 32-bit floating-point number and places the result in the specified result words. (The arc sine function is the inverse of the sine function; it returns the angle that produces a given sine value between –1 and 1.) 	Output	Required
ARC COSINE	ACOS @ACOS	464	 **S**: 1st source word **R**: 1st result word	Calculates the arc cosine of a 32-bit floating-point number and places the result in the specified result words. (The arc cosine function is the inverse of the cosine function; it returns the angle that produces a given cosine value between –1 and 1.) 	Output	Required
ARC TANGENT	ATAN @ATAN	465	 **S**: 1st source word **R**: 1st result word	Calculates the arc tangent of a 32-bit floating-point number and places the result in the specified result words. (The arc tangent function is the inverse of the tangent function; it returns the angle that produces a given tangent value.) 	Output	Required
SQUARE ROOT	SQRT @SQRT	466	 **S**: 1st source word **R**: 1st result word	Calculates the square root of a 32-bit floating-point number and places the result in the specified result words. 	Output	Required
EXPONENT	EXP @EXP	467	 **S**: 1st source word **R**: 1st result word	Calculates the natural (base e) exponential of a 32-bit floating-point number and places the result in the specified result words. 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LOGARITHM	LOG @LOG	468	 **S**: 1st source word **R**: 1st result word	Calculates the natural (base e) logarithm of a 32-bit floating-point number and places the result in the specified result words. 	Output	Required
EXPONENTIAL POWER	PWR @PWR	840	 **B**: 1st base word **E**: 1st exponent word **R**: 1st result word	Raises a 32-bit floating-point number to the power of another 32-bit floating-point number. 	Output	Required
FLOATING SYMBOL COMPARISON	LD, AND. or OR + =F, <>F, <F, <=F, >F, or >=F	329 (=F), 330 (<>F), 331 (<F), 332 (<=F), 333 (>F), 334 (>=F)	Using LD:  Using AND:  Using OR:  **S1**: Comparison data 1 **S2**: Comparison data 2	Compares the specified single-precision data (32 bits) or constants and creates an ON execution condition if the comparison result is true. Three kinds of symbols can be used with the floating-point symbol comparison instructions: LD (Load), AND, and OR.	LD: Logical start. AND or OR: Continues on rung	LD: Not required AND or OR: Required
FLOATING-POINT TO ASCII	FSTR @FSTR	448	 **S:** 1st source word **C**: Control word **D**: Destination word	Converts the specified single-precision floating-point data (32-bit decimal-point or exponential format) to text string data (ASCII) and outputs the result to the destination word.	Output	Required
ASCII TO FLOATING-POINT	FVAL @FVAL	449	 **S:** Source word **D**: 1st destination word	Converts the specified text string (ASCII) representation of single-precision floating-point data (decimal-point or exponential format) to 32-bit single-precision floating-point data and outputs the result to the destination words.	Output	Required

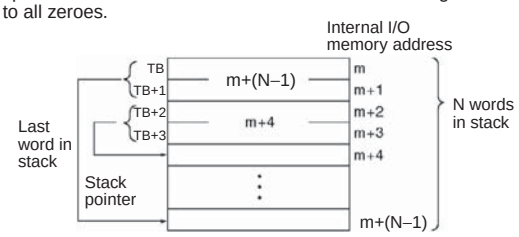
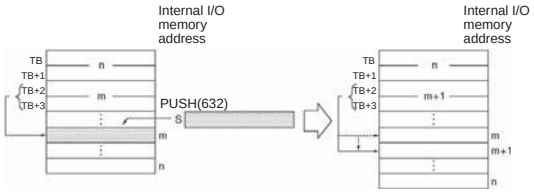
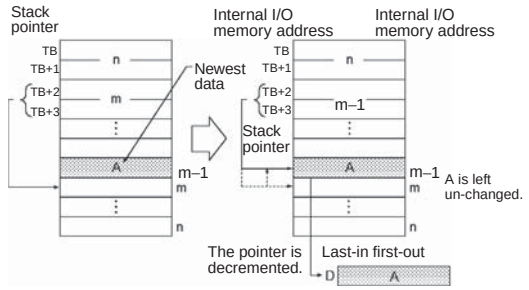
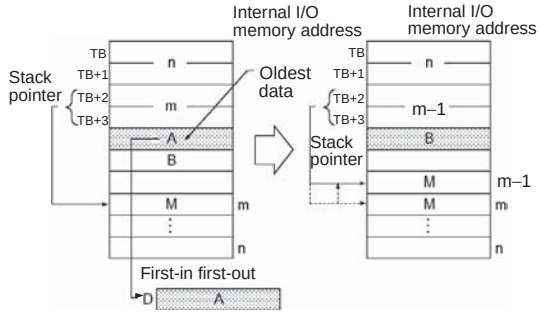
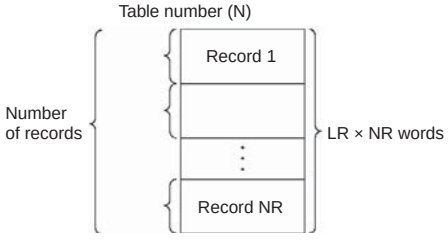
A-1-14 Double-precision Floating-point Instructions

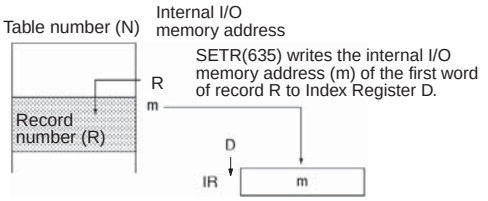
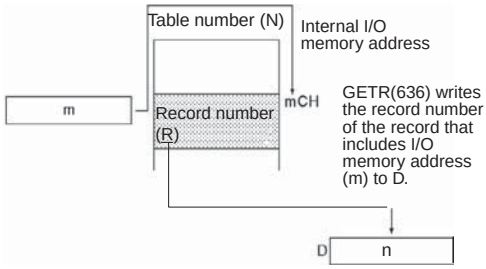
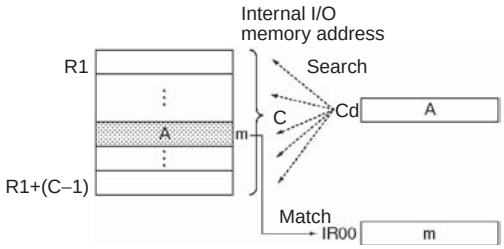
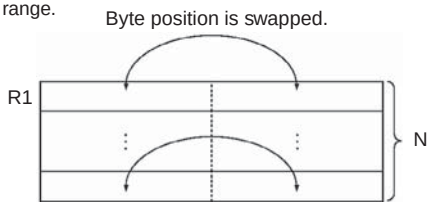
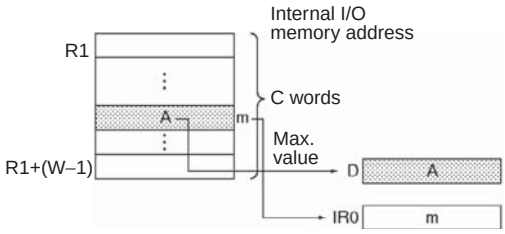
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE FLOATING TO 16-BIT BINARY	FIXD @FIXD	841	 **S:** 1st source word **D**: Destination word	Converts the specified double-precision floating-point data (64 bits) to 16-bit signed binary data and outputs the result to the destination word.	Output	Required
DOUBLE FLOATING TO 32-BIT BINARY	FIXLD @FIXLD	842	 **S:** 1st source word **D**: 1st destination word	Converts the specified double-precision floating-point data (64 bits) to 32-bit signed binary data and outputs the result to the destination words.	Output	Required
16-BIT BINARY TO DOUBLE FLOATING	DBL @DBL	843	 **S:** Source word **D**: 1st destination word	Converts the specified 16-bit signed binary data to double-precision floating-point data (64 bits) and outputs the result to the destination words.	Output	Required
32-BIT BINARY TO DOUBLE FLOATING	DBLL @DBLL	844	 **S:** 1st source word **D**: 1st destination word	Converts the specified 32-bit signed binary data to double-precision floating-point data (64 bits) and outputs the result to the destination words.	Output	Required
DOUBLE FLOATING-POINT ADD	+D @+D	845	 **Au:** 1st augend word **Ad**: 1st addend word **R**: 1st result word	Adds the specified double-precision floating-point values (64 bits each) and outputs the result to the result words.	Output	Required
DOUBLE FLOATING-POINT SUBTRACT	–D @–D	846	 **Mi:** 1st minuend word **Su**: 1st subtrahend word **R**: 1st result word	Subtracts the specified double-precision floating-point values (64 bits each) and outputs the result to the result words.	Output	Required
DOUBLE FLOATING-POINT MULTIPLY	*D @*D	847	 **Md:** 1st multiplicand word **Mr**: 1st multiplier word **R**: 1st result word	Multiplies the specified double-precision floating-point values (64 bits each) and outputs the result to the result words.	Output	Required

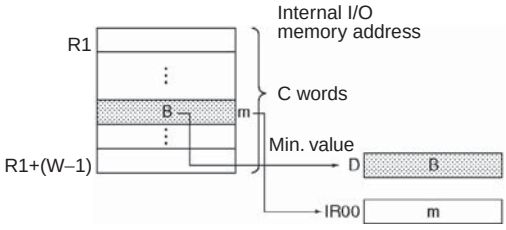
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE FLOATING-POINT DIVIDE	/D @/D	848	 **Dd:** 1st Dividend word **Dr**: 1st divisor word **R**: 1st result word	Divides the specified double-precision floating-point values (64 bits each) and outputs the result to the result words.	Output	Required
DOUBLE DEGREES TO RADIANS	RADD @RADD	849	 **S:** 1st source word **R**: 1st result word	Converts the specified double-precision floating-point data (64 bits) from degrees to radians and outputs the result to the result words.	Output	Required
DOUBLE RADIANS TO DEGREES	DEGD @DEGD	850	 **S:** 1st source word **R**: 1st result word	Converts the specified double-precision floating-point data (64 bits) from radians to degrees and outputs the result to the result words.	Output	Required
DOUBLE SINE	SIND @SIND	851	 **S:** 1st source word **R**: 1st result word	Calculates the sine of the angle (radians) in the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE COSINE	COSD @COSD	852	 **S:** 1st source word **R**: 1st result word	Calculates the cosine of the angle (radians) in the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE TANGENT	TAND @TAND	853	 **S:** 1st source word **R**: 1st result word	Calculates the tangent of the angle (radians) in the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE ARC SINE	ASIND @ASIND	854	 **S:** 1st source word **R**: 1st result word	Calculates the angle (in radians) from the sine value in the specified double-precision floating-point data (64 bits) and outputs the result to the result words. (The arc sine function is the inverse of the sine function; it returns the angle that produces a given sine value between −1 and 1.)	Output	Required
DOUBLE ARC COSINE	ACOSD @ACOSD	855	 **S:** 1st source word **R**: 1st result word	Calculates the angle (in radians) from the cosine value in the specified double-precision floating-point data (64 bits) and outputs the result to the result words. (The arc cosine function is the inverse of the cosine function; it returns the angle that produces a given cosine value between −1 and 1.)	Output	Required

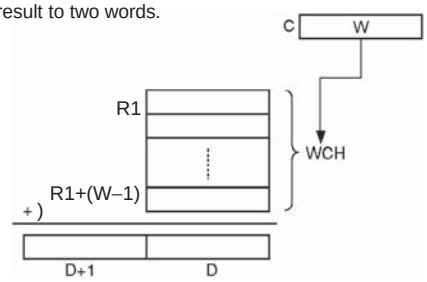
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DOUBLE ARC TANGENT	ATAND @ATAND	856	 **S:** 1st source word **R**: 1st result word	Calculates the angle (in radians) from the tangent value in the specified double-precision floating-point data (64 bits) and outputs the result to the result words. (The arc tangent function is the inverse of the tangent function; it returns the angle that produces a given tangent value.)	Output	Required
DOUBLE SQUARE ROOT	SQRTD @SQRTD	857	 **S:** 1st source word **R**: 1st result word	Calculates the square root of the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE EXPONENT	EXPD @EXPD	858	 **S:** 1st source word **R**: 1st result word	Calculates the natural (base e) exponential of the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE LOGARITHM	LOGD @LOGD	859	 **S:** 1st source word **R**: 1st result word	Calculates the natural (base e) logarithm of the specified double-precision floating-point data (64 bits) and outputs the result to the result words.	Output	Required
DOUBLE EXPONENTIAL POWER	PWRD @PWRD	860	 **B:** 1st base word **E**: 1st exponent word **R**: 1st result word	Raises a double-precision floating-point number (64 bits) to the power of another double-precision floating-point number and outputs the result to the result words.	Output	Required
DOUBLE SYMBOL COMPARISON	LD, AND. or OR + =D, <>D, <D, <=D, >D, or >=D	LD, AND. or OR + 335 (=D), 336 (<>D), 337 (<D), 338 (<=D), 339 (>D), 340 (>=D)	Using LD:  Using AND:  Using OR:  **S1**: Comparison data 1 **S2**: Comparison data 2	Compares the specified double-precision data (64 bits) and creates an ON execution condition if the comparison result is true. Three kinds of symbols can be used with the floating-point symbol comparison instructions: LD (Load), AND, and OR.	LD: Logical start. AND or OR: Continues on rung	LD: Not required AND or OR: Required

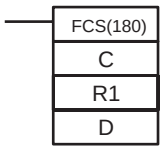
A-1-15 Table Data Processing Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SET STACK	SSET @SSET	630	SSET(630) TB N **TB**: 1st stack address **N**: Number of words	Defines a stack of the specified length beginning at the specified word and initializes the words in the data region to all zeroes. 	Output	Required
PUSH ONTO STACK	PUSH @PUSH	632	PUSH(632) TB S **TB**: 1st stack address **S**: Source word	Writes one word of data to the specified stack. 	Output	Required
LAST IN FIRST OUT	LIFO @LIFO	634	LIFO(634) TB D **TB**: 1st stack address **D**: Destination word	Reads the last word of data written to the specified stack (the newest data in the stack). 	Output	Required
FIRST IN FIRST OUT	FIFO @FIFO	633	FIFO(633) TB D **TB**: 1st stack address **D**: Destination word	Reads the first word of data written to the specified stack (the oldest data in the stack). 	Output	Required
DIMENSION RECORD TABLE	DIM @DIM	631	DIM(631) N LR NR TB **N**: Table number **LR**: Length of each record **NR**: Number of records **TB**: 1st table word	Defines a record table by declaring the length of each record and the number of records. Up to 16 record tables can be defined. 	Output	Required

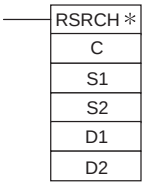
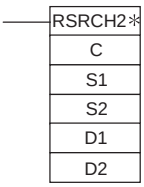
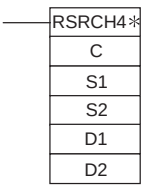
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SET RECORD LOCATION	SETR @SETR	635	SETR(635) N R D **N**: Table number **R**: Record number **D**: Destination Index Register	Writes the location of the specified record (the internal I/O memory address of the beginning of the record) in the specified Index Register. 	Output	Required
GET RECORD NUMBER	GETR @GETR	636	GETR(636) N IR D **N**: Table number **IR**: Index Register **D**: Destination word	Returns the record number of the record at the internal I/O memory address contained in the specified Index Register. 	Output	Required
DATA SEARCH	SRCH @SRCH	181	SRCH(181) C R1 Cd **C**: 1st control word **R1**: 1st word in range **Cd**: Comparison data	Searches for a word of data within a range of words. 	Output	Required
SWAP BYTES	SWAP BYTES	637	SWAP(637) N R1 **N**: Number of words **R1**: 1st word in range	Switches the leftmost and rightmost bytes in all of the words in the range. 	Output	Required
FIND MAXIMUM	MAX @MAX	182	MAX(182) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: Destination word	Finds the maximum value in the range. 	Output	Required
DOUBLE FIND MAXIMUM	MAXL @MAXL	174	MAXL(174) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as double-word table data beginning from the first word in the range specified by R1, and outputs the maximum value in the table to D+1 and D.	Output	Required

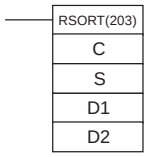
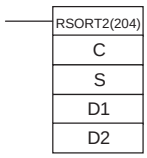
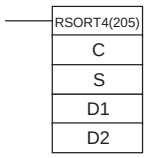
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FIND MAXIMUM FLOATING	MAXF @MAXF	176	MAXF(176) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as a table of single-precision floating-point data (double-word data) beginning from the first word in the range specified by R1, and outputs the maximum value in the table to D+1 and D.	Output	Required
FIND DOUBLE MAXIMUM FLOATING	MAXD @MAXD	178	MAXD(178) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as a table of double-precision floating-point data (double-word data) beginning from the first word in the range specified by R1, and outputs the maximum value in the table to D+1 and D.	Output	Required
FIND MINIMUM	MIN @MIN	183	MIN(183) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: Destination word	Finds the minimum value in the range.  	Output	Required
DOUBLE FIND MINIMUM	MINL @MINL	175	MINL(175) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as double-word table data beginning from the first word in the range specified by R1, and outputs the minimum value in the table to D+1 and D.	Output	Required
FIND MINIMUM FLOATING	MINF @MINF	177	MINF(177) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as a table of single-precision floating-point data (double-word data) beginning from the first word in the range specified by R1, and outputs the minimum value in the table to D+1 and D.	Output	Required
FIND DOUBLE MINIMUM FLOATING	MIND @MIND	179	MIND(179) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Treats the number of data items specified by C as a table of double-precision floating-point data (double-word data) beginning from the first word in the range specified by R1, and outputs the minimum value in the table to D+1 and D.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SUM	SUM @SUM	184	SUM(184) C R1 D **C**: 1st control word **R1**: 1st word in range **D**: 1st destination word	Adds the bytes or words in the range and outputs the result to two words. 	Output	Required
STACK SIZE READ	SNUM @SNUM	638	SNUM(638) TB D **TB**: First stack address **D**: Destination word	Counts the amount of stack data (number of words) in the specified stack.	Output	Required
STACK DATA READ	SREAD @SREAD	639	SREAD(639) TB C D **TB**: First stack address **C**: Offset value **D**: Destination word	Reads the data from the specified data element in the stack. The offset value indicates the location of the desired data element (how many data elements before the current pointer position).	Output	Required
STACK DATA OVERWRITE	SWRIT @SWRIT	640	SWRIT(640) TB C S **TB**: First stack address **C**: Offset value **S**: Source data	Writes the source data to the specified data element in the stack (overwriting the existing data). The offset value indicates the location of the desired data element (how many data elements before the current pointer position).	Output	Required
STACK DATA INSERT	SINS @SINS	641	SINS(641) TB C S **TB**: First stack address **C**: Offset value **S**: Source data	Inserts the source data at the specified location in the stack and shifts the rest of the data in the stack downward. The offset value indicates the location of the insertion point (how many data elements before the current pointer position).	Output	Required
STACK DATA DELETE	SDEL @SDEL	642	SDEL(642) TB C D **TB**: First stack address **C**: Offset value **D**: Destination word	Deletes the data element at the specified location in the stack and shifts the rest of the data in the stack upward. The offset value indicates the location of the deletion point (how many data elements before the current pointer position).	Output	Required

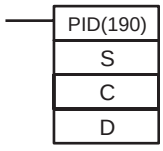
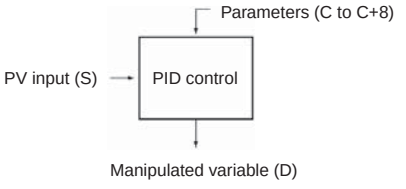
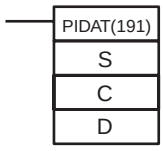
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FRAME CHECKSUM	FCS @FCS	180	 **C**: 1st control word **R1:** 1st word in range **D**: 1st destination word	Calculates the FCS value for the specified range and outputs the value in ASCII.	Output	Required

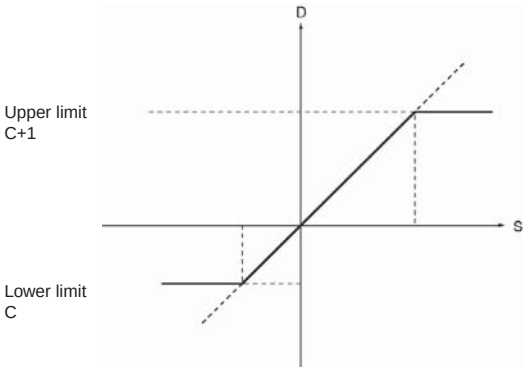
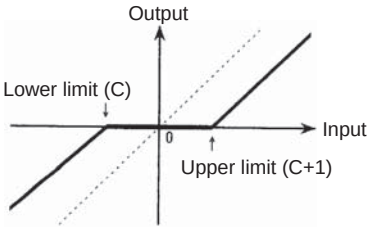
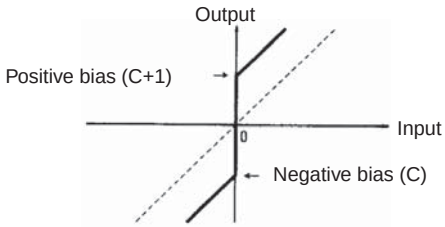
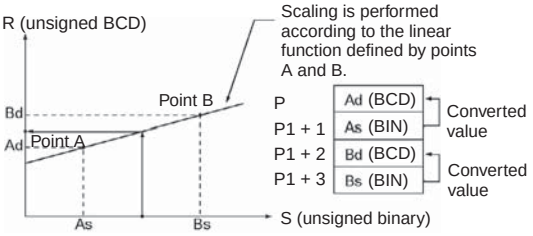
A-1-16 Tracking Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
Unsigned One-word Record Search Instructions	RSRCH + <, <=, =, >, >= @RSRCH + <, <=, =, >, >=	360 (<), 361 (<=), 362 (=), 363 (>), 364 (>=)	 **C**: 1st control word **S1:** 1st word of first record to search **S2:** Search data **D1:** 1st destination word **D2:** Destination index register	An Unsigned One-word Record Search Instruction searches the data (1 word) specified by S2, beginning from the table specified by S1. When a record matching the specified condition is found, its record number and data are output to D1 onwards. When an index register is specified for D2, the address of the matching record is output to that index register. To not use an index register output, set #00000000 for D2.	Output	Required
Unsigned Two-word Record Search Instructions	RSRCH2 + <, <=, =, >, >= @RSRCH2 + <, <=, =, >, >=	370 (<), 371 (<=), 372 (=), 373 (>), 374 (>=)	 **C:** 1st control word **S1:** 1st word of first record to search **S2:** 1st word of search data **D1:** 1st destination word **D2:** Destination index register	An Unsigned Two-word Record Search Instruction searches the data (2 words) specified by S2, beginning from the table specified by S1. When a record matching the specified condition is found, its record number and data are output to D1 onwards. When an index register is specified for D2, the address of the matching record is output to that index register. To not use an index register output, set #00000000 for D2.	Output	Required
Unsigned Four-word Record Search Instructions	RSRCH4 + <, <=, =, >, >= @RSRCH4 + <, <=, =, >, >=	380 (<), 381 (<=), 382 (=), 383 (>), 384 (>=)	 **C:** 1st control word **S1:** 1st word of first record to search **S2:** Search data **D1:** 1st destination word **D2:** Destination index register	An Unsigned Four-word Record Search Instruction searches the data (4 words) specified by S2, beginning from the table specified by S1. When a record matching the specified condition is found, its record number and data are output to D1 onwards. When an index register is specified for D2, the address of the matching record is output to that index register. To not use an index register output, set #00000000 for D2.	Output	Required

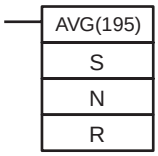
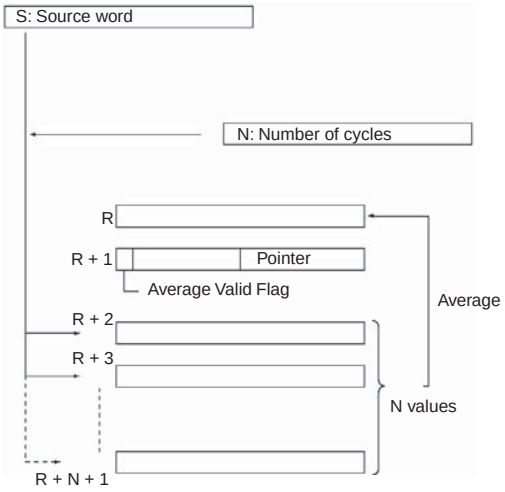
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
UNSIGNED ONE-WORD RECORD SORT	RSORT @RSORT	203	 **C:** 1st control word **S:** 1st word of first record to sort **D1:** 1st word of sorting results **D2:** Destination index register	Sorts the records (1 word) in the table specified by S, according to the control words.	Output	Required
UNSIGNED TWO-WORD RECORD SORT	RSORT2 @RSORT2	204	 **C:** 1st control word **S:** 1st word of first record to sort **D1:** 1st word of sorting results **D2:** Destination index register	Sorts the records (2 words) in the table specified by S, according to the control words.	Output	Required
UNSIGNED FOUR-WORD RECORD SORT	RSORT4 @RSORT4	205	 **C:** 1st control word **S:** 1st word of first record to sort **D1:** 1st word of sorting results **D2:** Destination index register	Sorts the records (4 words) in the table specified by S, according to the control words.	Output	Required

A-1-17 Data Control Instructions

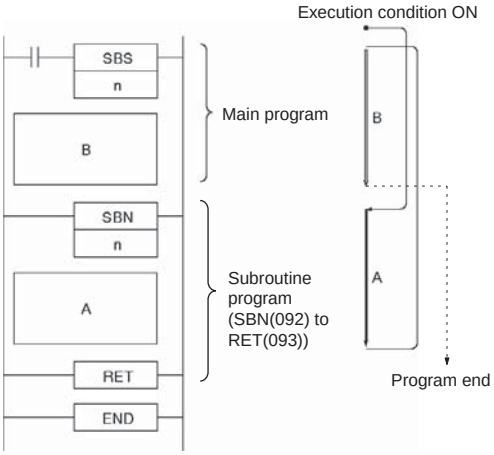
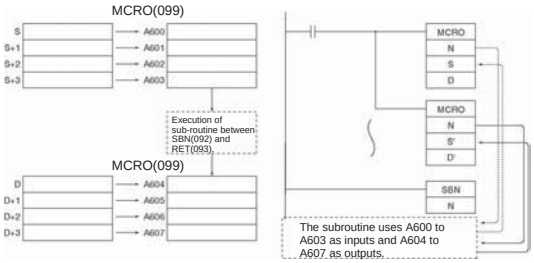
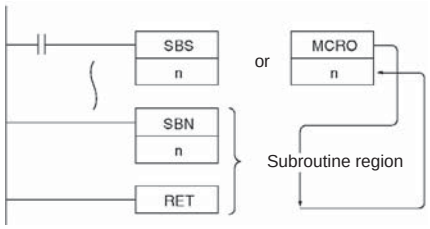
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
PID CONTROL	PID	190	 **S**: Input word **C**: 1st parameter word **D**: Output word	Executes PID control according to the specified parameters. 	Output	Required
PID CONTROL WITH AUTOTUNING	PIDAT	191	 **S**: Input word **C**: 1st parameter word **D**: Output word	Executes PID control according to the specified parameters. The PID constants can be auto-tuned with PIDAT(191).	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LIMIT CONTROL	LMT @LMT	680	LMT(680) S C D **S**: Input word **C**: 1st limit word **D**: Output word	Controls output data according to whether or not input data is within upper and lower limits. 	Output	Required
DEAD BAND CONTROL	BAND @BAND	681	BAND(681) S C D **S**: Input word **C**: 1st limit word **D**: Output word	Controls output data according to whether or not input data is within the dead band range. 	Output	Required
DEAD ZONE CONTROL	ZONE @ZONE	682	ZONE(682) S C D **S**: Input word **C**: 1st limit word **D**: Output word	Adds the specified bias to input data and outputs the result. 	Output	Required
TIME-PROPORTIONAL OUTPUT	TPO	685	TPO (685) S C R **S**: Input word **C**: 1st parameter word **R**: Pulse Output Bit	Inputs the duty ratio or manipulated variable from the specified word, converts the duty ratio to a time-proportional output based on the specified parameters, and outputs the result from the specified output.	Output	Required
SCALING	SCL @SCL	194	SCL(194) S P1 R **S**: Source word **P1**: 1st parameter word **R**: Result word	Converts unsigned binary data into unsigned BCD data according to the specified linear function. 	Output	Required

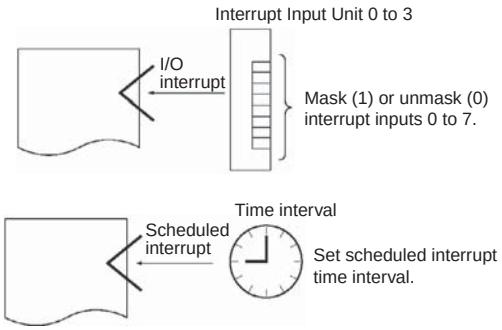
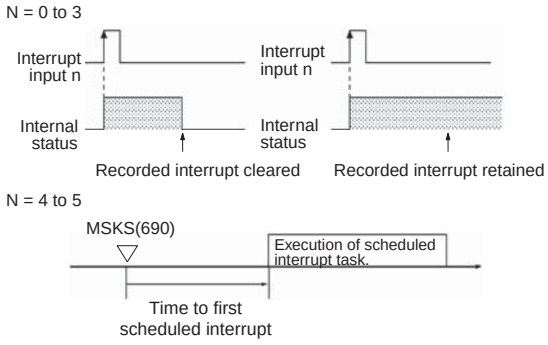
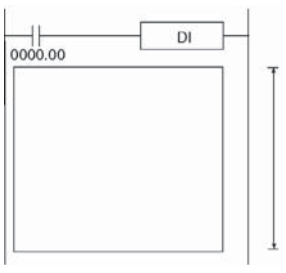
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SCALING 2	SCL2 @SCL2	486	SCL2(486) S P1 R **S**: Source word **P1**: 1st parameter word **R**: Result word	Converts signed binary data into signed BCD data according to the specified linear function. An offset can be input in defining the linear function. **Positive Offset** R (signed BCD) Offset ΔY ΔX S (signed binary) **Negative Offset** R (signed BCD) ΔY ΔX Offset S (signed binary) P1 Offset (Signed binary) P1 + 1 ΔY (Signed binary) P1 + 2 ΔX (Signed BCD) **Offset of 0000** R (signed BCD) Offset = 0000 hex ΔY ΔX S (signed binary)	Output	Required
SCALING 3	SCL3 @SCL3	487	SCL3(487) S P1 R **S**: Source word **P1**: 1st parameter word **R**: Result word	Converts signed BCD data into signed binary data according to the specified linear function. An offset can be input in defining the linear function. **Positive Offset** R (signed binary) Max. conversion ΔY ΔX Offset Min. conversion S (signed BCD) **Negative Offset** R (signed binary) Max conversion ΔY ΔX Offset S (signed BCD) Min. conversion **Offset of 0000** R (signed binary) Max. conversion ΔY ΔX S (signed BCD) Min. conversion	Output	Required

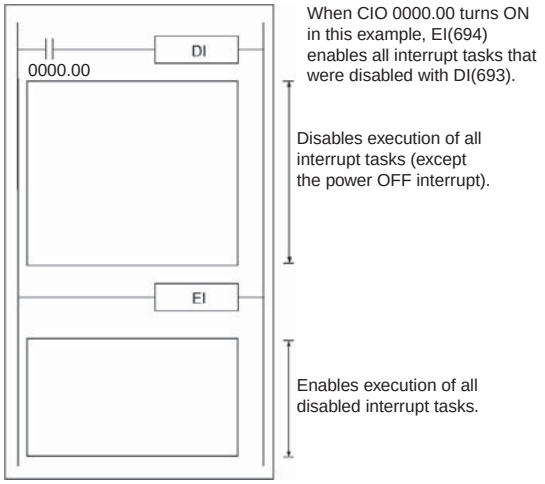
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
AVERAGE	AVG	195	 **S**: Source word **N**: Number of cycles **R**: Result word	Calculates the average value of an input word for the specified number of cycles. 	Output	Required

A-1-18 Subroutine Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SUBROUTINE CALL	SBS @SBS	091	SBS(091) N **N**: Subroutine number	Calls the subroutine with the specified subroutine number and executes that program. 	Output	Required
MACRO	MCRO @MCRO	099	MCRO(099) N S D **N**: Subroutine number **S**: 1st input parameter word **D**: 1st output parameter word	Calls the subroutine with the specified subroutine number and executes that program using the input parameters in S to S+3 and the output parameters in D to D+3. 	Output	Required
SUBROUTINE ENTRY	SBN	092	SBN(092) N **N**: Subroutine number	Indicates the beginning of the subroutine program with the specified subroutine number. 	Output	Required
SUBROUTINE RETURN	RET	093	RET(093)	Indicates the end of a subroutine program.	Output	Not required
GLOBAL SUBROUTINE CALL	GSBS @GSBS	750	GSBS(750) N **N**: Subroutine number	Calls the subroutine with the specified subroutine number and executes that program.	Output	Required
GLOBAL SUBROUTINE ENTRY	GSBN	751	GSBN(751) N **N**: Subroutine number	Indicates the beginning of the subroutine program with the specified subroutine number.	Output	Not required
GLOBAL SUBROUTINE RETURN	GRET	752	GRET(752)	Indicates the end of a subroutine program.	Output	Not required

A-1-19 Interrupt Control Instructions

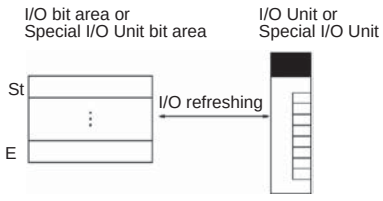
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SET INTERRUPT MASK	MSKS @MSKS	690	MSKS(690) N C **N**: Interrupt number **C**: Control data	Sets up interrupt processing for I/O interrupts or scheduled interrupts. Both I/O interrupt tasks and scheduled interrupt tasks are masked (disabled) when the PC is first turned on. MSKS(690) can be used to unmask or mask I/O interrupts and set the time intervals for scheduled interrupts. 	Output	Required
READ INTERRUPT MASK	MSKR @MSKR	692	MSKR(692) N D **N**: Interrupt number **D**: Destination word	Reads the current interrupt processing settings that were set with MSKS(690).	Output	Required
CLEAR INTERRUPT	CLI @CLI	691	CLI(691) N C **N**: Interrupt number **C**: Control data	Clears or retains recorded interrupt inputs for I/O interrupts or sets the time to the first scheduled interrupt for scheduled interrupts. 	Output	Required
DISABLE INTERRUPTS	DI @DI	693	DI(693)	Disables execution of all interrupt tasks except the power OFF interrupt.  When CIO 0000.00 turns ON in this example, all interrupt tasks except for the power OFF interrupt task are disabled. Disables execution of all interrupt tasks (except the power OFF interrupt).	Output	Required

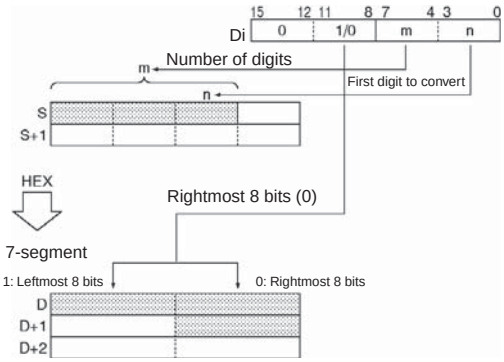
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
ENABLE INTER-RUPTS	EI	694	EI(694)	Enables execution of all interrupt tasks that were disabled with DI(693). 	Output	Not required

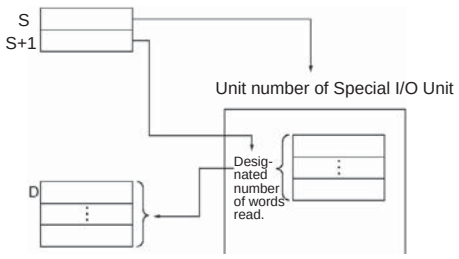
A-1-20 Step Instructions

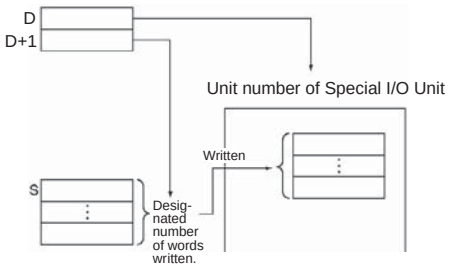
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
STEP DEFINE	STEP	008	STEP(008) B **B**: Bit	STEP(008) functions in following 2 ways, depending on its position and whether or not a control bit has been specified. (1) Starts a specific step. (2) Ends the step programming area (i.e., step execution). The step program area is from the first STEP(008) instruction (which always takes a control bit) to the last STEP(008) instruction (which never takes a control bit).	Output	Required
STEP START	SNXT	009	SNXT(009) B **B**: Bit	SNXT(009) is used in the following three ways: (1) To start step programming execution. (2) To proceed to the next step control bit. (3) To end step programming execution.	Output	Required

A-1-21 Basic I/O Unit Instructions

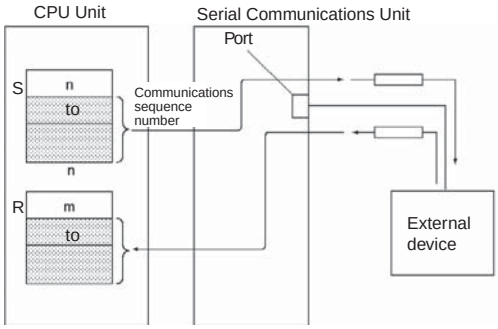
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
I/O REFRESH	IORF @IORF	097	IORF(097) St E **St**: Starting word **E**: End word	Refreshes the specified I/O words. 	Output	Required
SPECIAL I/O UNIT I/O REFRESH	FIORF @FIORF	225	FIORF(225) N **N**: Unit number	Immediately refreshes the I/O words allocated to the Special I/O Unit with the specified unit number.	Output	Required
CPU BUS UNIT I/O REFRESH	DLNK @DLNK	226	DLNK(226) N **N**: Unit number	Immediately refreshes the I/O words allocated to the CPU Bus Unit with the specified unit number.	Output	Required

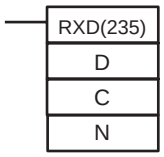
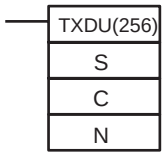
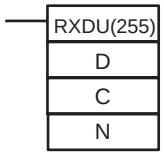
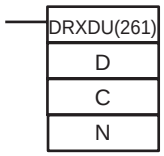
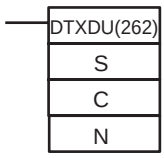
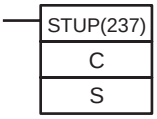
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
7-SEGMENT DECODER	SDEC @SDEC	078	SDEC(078) S Di D **S**: Source word **Di**: Digit designator **D**: 1st destination word	Converts the hexadecimal contents of the designated digit(s) into 8-bit, 7-segment display code and places it into the upper or lower 8-bits of the specified destination words. 	Output	Required
DIGITAL SWITCH INPUT	DSW	210	DSW (210) I O D C1 C2 **I**: Data input word (D0 to D3) **O**: Output word **D**: 1st result word **C1**: Number of digits **C2**: System word	Reads the value set on an external digital switch (or thumbwheel switch) connected to an Input Unit or Output Unit and stores the 4-digit or 8-digit BCD data in the specified words.	Output	Required
TEN KEY INPUT	TKY	211	TKY (211) I D1 D2 **I**: Data input word **D1**: 1st register word **D2**: Key input word	Reads numeric data from a ten-key keypad connected to an Input Unit and stores up to 8 digits of BCD data in the specified words.	Output	Required
HEXADECIMAL KEY INPUT	HKY	212	HKY (212) I O D C **I**: Data input word **O**: Output word **D**: 1st register word **C**: System word	Reads numeric data from a hexadecimal keypad connected to an Input Unit and Output Unit and stores up to 8 digits of hexadecimal data in the specified words.	Output	Required
MATRIX INPUT	MTR	213	MTR (213) I O D C **I**: Data input word **O**: Output word **D**: 1st destination word **C**: System word	Inputs up to 64 signals from an 8 × 8 matrix connected to an Input Unit and Output Unit (using 8 input points and 8 output points) and stores that 64-bit data in the 4 destination words.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
7-SEGMENT DISPLAY OUTPUT	7SEG	214	7SEG(214) S O C D **S**: 1st source word **O**: Output word **C**: Control data **D**: System word	Converts the source data (either 4-digit or 8-digit BCD) to 7-segment display data, and outputs that data to the specified output word.	Output	Required
ANALOG INPUT DIRECT CONVERSION (for CJ1W-AD042)	AIDC @AIDC	216	AIDC(216) N A **N**: Unit number **A**: Analog input number	Reads the input conversion value of the specified analog input number from the CJ1W-AD042 Analog Input Unit in Direct Conversion Mode.	Output	Required
ANALOG OUTPUT DIRECT CONVERSION (for CJ1W-DA042V)	AODC @AODC	217	AODC(217) N A **N**: Unit number **A**: Analog output number	Outputs the output set value for the specified analog output number to the CJ1W-DA042V Analog Output Unit in Direct Conversion Mode.	Output	Required
PCU HIGH-SPEED POSITIONING (CJ1W-NC□□4 or CJ1W-NC□81 only)	NCDMV @NCDMV	218	NCDMV(218) C A **C:** Control data **A:** First word of Direct Operation Command Area	NCDMV(218) starts absolute or relative high-speed point-to-point positioning for the specified axis of a CJ1W-NC□□4 or CJ1W-NC□81 Position Control Unit.	Output	Required
PCU POSITIONING TRIGGER (CJ1W-NC□81 only)	NCDTR @NCDTR	219	NCDTR(219) C **C**: Control data	NCDTR(219) is used to start a sequence for Memory Operation of a CJ1W-NC□81 Position Control Unit when the start condition for the sequence is waiting for a command from NCDTR(219).	Output	Required
INTELLIGENT I/O READ	IORD @IORD	222	IORD(222) C S D **C**: Control data **S**: Transfer source and number of words **D**: Transfer destination and number of words	Reads the contents of the memory area for the Special I/O Unit or CPU Bus Unit (see note). 	Output	Required

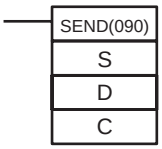
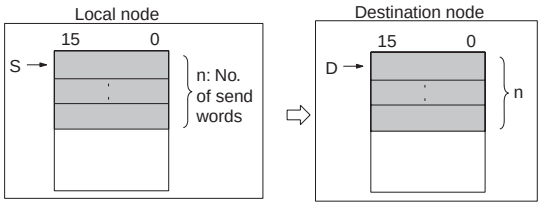
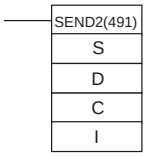
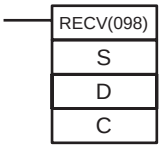
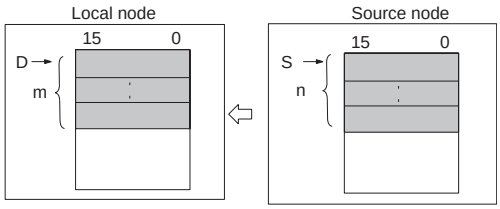
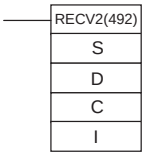
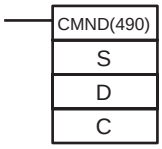
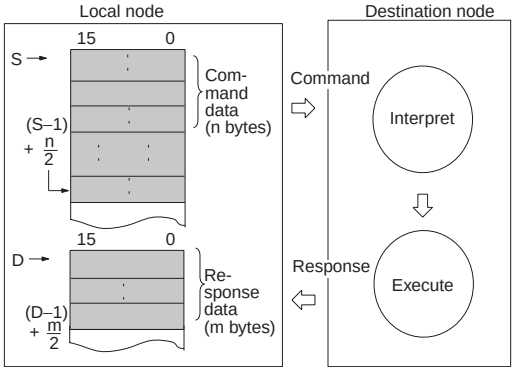
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
INTELLI-GENT I/O WRITE	IOWR @IOWR	223	IOWR(223) C S D **C**: Control data **S**: Transfer source and number of words **D**: Transfer destination and number of words	Outputs the contents of the CPU Unit's I/O memory area to the Special I/O Unit or the CPU Bus Unit (see note). 	Output	Required

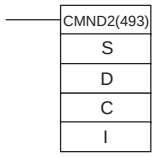
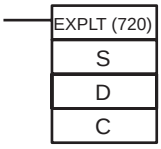
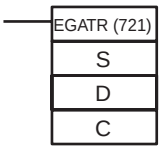
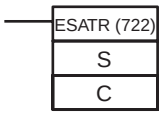
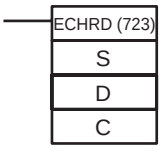
A-1-22 Serial Communications Instructions

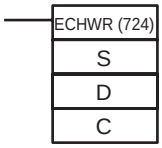
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
PROTOCOL MACRO	PMCR @PMCR	260	PMCR(260) C1 C2 S R **C1**: Control word 1 **C2**: Control word 2 **S**: 1st send word **R**: 1st receive word	Calls and executes a communications sequence registered in a Serial Communications Board (CS Series only) or Serial Communications Unit. 	Output	Required
PROTOCOL-MACRO 2	PMCR2 @PMCR2	264	PMCR2(264) C1 C2 S R I **C1:** Control word 1 **C2:** Control word 2 **S:** 1st send word **R:** 1st receive word **I:** 1st communications information word	Up to 64 PMCR2 instructions can be used simultaneously. They are otherwise the same as PMCR.	Output	Required
TRANSMIT	TXD @TXD	236	TXD(236) S C N **S**: 1st source word **C**: Control word **N**: Number of bytes 0000 to 0100 hex (0 to 256 decimal)	Outputs the specified number of bytes of data without conversion from the RS-232C port built into the CPU Unit (no-protocol mode) or the serial port of a Serial Communications Board or Unit with unit version 1.2 or later (no-protocol mode) according to the start code and end code specified for no-protocol mode in the PLC Setup.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
RECEIVE	RXD @RXD	235	 **D**: 1st destination word **C**: Control word **N**: Number of bytes to store 0000 to 0100 hex (0 to 256 decimal)	Reads the specified number of bytes of data starting with the specified first word from the RS-232C port built into the CPU Unit (no-protocol mode) or the serial port of a Serial Communications Board or Unit with unit version 1.2 or later (no-protocol mode) according to the start code and end code specified for no-protocol mode in the PLC Setup.	Output	Required
TRANSMIT VIA SERIAL COMMUNICATIONS UNIT	TXDU @TXDU	256	 **S**: 1st source word **C**: 1st control word **N**: Number of bytes (0000 to 0256 BCD)	Outputs the specified number of bytes of data without conversion from the serial port of a Serial Communications Unit with unit version 1.2 or later. The data is output in no-protocol mode with the start code and end code (if any) specified in the allocated DM Area.	Output	Required
RECEIVE VIA SERIAL COMMUNICATIONS UNIT	RXDU @RXDU	255	 **D**: 1st destination word **C**: 1st control word **N**: Number of bytes to store (0000 to 0100 hex)	Reads the specified number of bytes of data starting with the specified first word from the serial port of a Serial Communications Unit with unit version 1.2 or later. The data is read in no-protocol mode with the start code and end code (if any) specified in the allocated DM Setup Area.	Output	Required
DIRECT RECEIVE VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (CJ1W-SCU22, CJ1W-SCU32, or CJ1W-SCU42 only)	DRXDU @DRXDU	261	 **D**: 1st destination word **C**: 1st control word **N**: Number of bytes 0000 to 0100 hex (0 to 256)	Reads the specified number of bytes of data from the serial port of a CJ1W-SCU22/SCU32/SCU42 Serial Communications Unit to CPU Unit memory starting at the specified first word. The data is read in no-protocol mode with the start code and end code (if any) specified in the allocated DM Setup Area. This instruction reads the data from the Serial Communications Unit as soon as the instruction is executed to achieve high-speed data reception.	Output	Required
DIRECT TRANSMIT VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (CJ1W-SCU22, CJ1W-SCU32, or CJ1W-SCU42 only)	DTXDU @DTXDI	262	 **S**: 1st source word **C**: 1st control word **N**: Number of bytes 0000 to 0100 hex (0 to 256)	Outputs the specified number of bytes of data from the serial port of a CJ1WSCU22/SCU32/SCU42 Serial Communications Unit. The data is output in no-protocol mode from the specified first word with the start code and end code (if any) specified in the allocated DM Setup Area. This instruction sends the data to the Serial Communications Unit as soon as the instruction is executed to achieve high-speed data transmission.	Output	Required
CHANGE SERIAL PORT SETUP	STUP @STUP	237	 **C**: Control word (port) **S**: First source word	Changes the communications parameters of a serial port on the CPU Unit, Serial Communications Unit (CPU Bus Unit), or Serial Communications Board. STUP(237) thus enables the protocol mode to be changed during PLC operation.	Output	Required

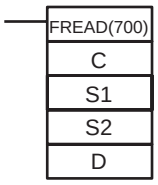
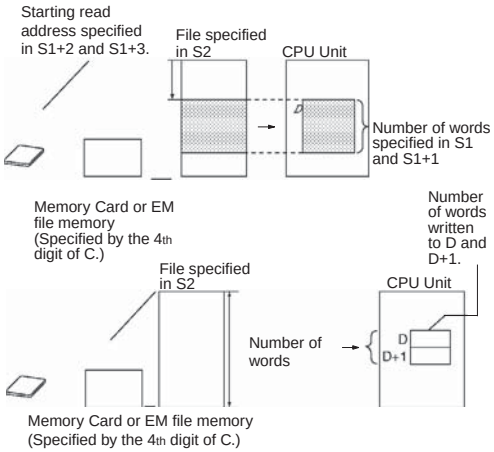
A-1-23 Network Instructions

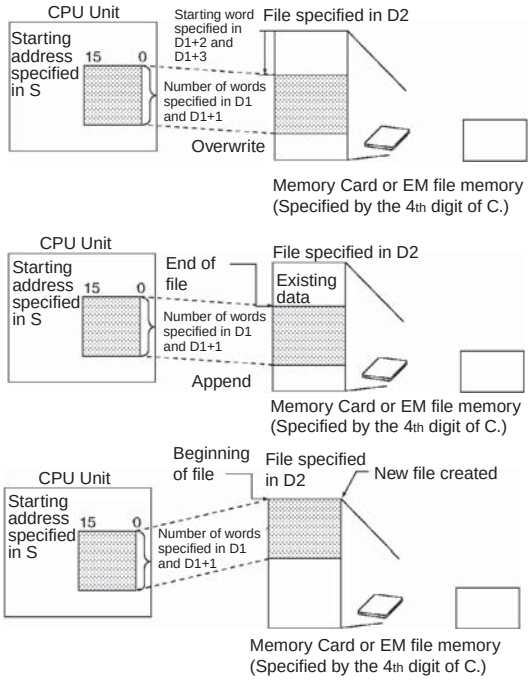
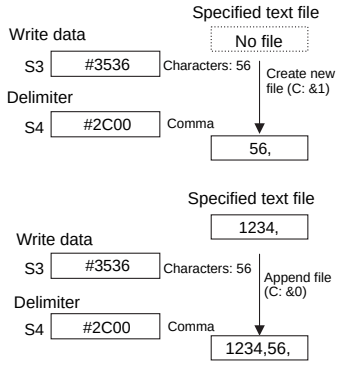
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
NETWORK SEND	SEND @SEND	090	 **S**: 1st source word **D**: 1st destination word **C**: 1st control word	Transmits data to a node in the network. 	Output	Required
NETWORK SEND 2	SEND2 @DEND2	491	 **S:** 1st source word **D:** 1st destination word **C:** 1st control word **I:** 1st communications information word	Up to 64 SEND2 instructions can be used simultaneously. They are otherwise the same as SEND.	Output	Required
NETWORK RECEIVE	RECV @RECV	098	 **S**: 1st source word **D**: 1st destination word **C**: 1st control word	Requests data to be transmitted from a node in the network and receives the data. 	Output	Required
NETWORK RECEIVE 2	RECV2 @RECV2	492	 **S:** 1st source word **D:** 1st destination word **C:** 1st control word **I:** 1st communications information word	Up to 64 RECV2 instructions can be used simultaneously. They are otherwise the same as RECV.	Output	Required
DELIVER COMMAND	CMND @CMND	490	 **S**: 1st command word **D**: 1st response word **C**: 1st control word	Sends FINS commands and receives the response 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DELIVER COMMAND 2	CMND2 @CMND2	493	 **S:** 1st command word **D:** 1st response word **C:** 1st control word **I:** 1st communications information word	Up to 64 CMND2 instructions can be used simultaneously. They are otherwise the same as CMND.	Output	Required
EXPLICIT MESSAGE SEND	EXPLT	720	 **S**: 1st word of send message **D**: 1st word of received message **C**: 1st control word	Sends an explicit message with any Service Code.	Output	Required
EXPLICIT GET ATTRIBUTE	EGATR	721	 **S**: 1st word of send message **D**: 1st word of received message **C**: 1st control word message	Reads status information with an explicit message (Get Attribute Single, Service Code: 0E hex).	Output	Required
EXPLICIT SET ATTRIBUTE	ESATR	722	 **S**: First word of send message **C**: First control word	Writes status information with an explicit message (Set Attribute Single, Service Code: 0E hex)	Output	Required
EXPLICIT WORD READ	ECHRD	723	 **S**: 1st source word in remote CPU Unit **D**: 1st destination word in local CPU Unit **C**: 1st control word	Reads data to the local CPU Unit from a remote CPU Unit in the network. (The remote CPU Unit must support explicit messages.)	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
EXPLICIT WORD WRITE	ECHWR	724	 **S**: 1st source word in local CPU Unit **D**: 1st destination word in remote CPU Unit **C**: 1st control word	Writes data from the local CPU Unit to a remote CPU Unit in the network. (The remote CPU Unit must support explicit messages.)	Output	Required

A-1-24 File Memory Instructions

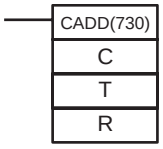
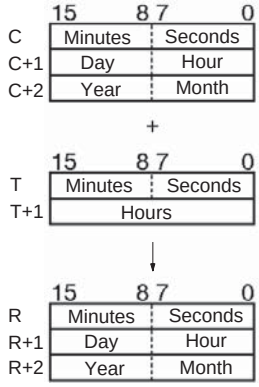
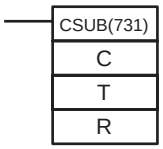
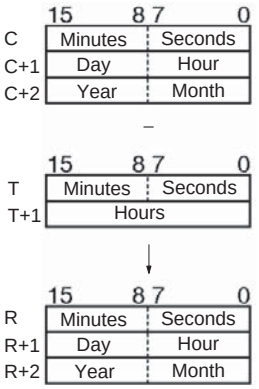
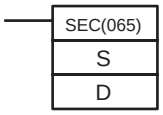
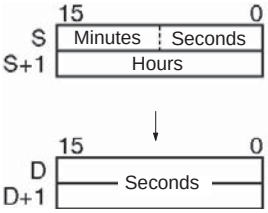
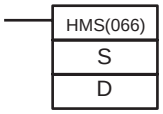
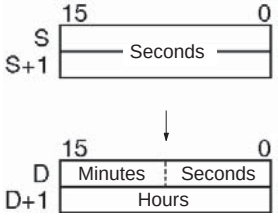
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
READ DATA FILE	FREAD @FREAD	700	 **C**: Control word **S1**: 1st source word **S2**: Filename **D**: 1st destination word	Reads the specified data or amount of data from the specified data file in file memory to the specified data area in the CPU Unit. 	Output	Required

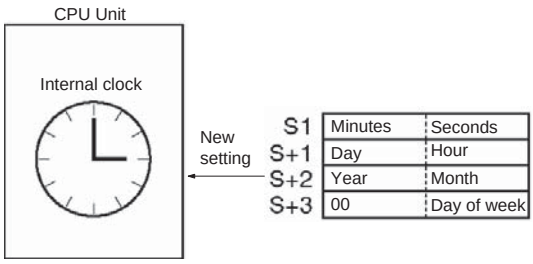
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
WRITE DATA FILE	FWRIT @FWRIT	701	FWRIT(701) C D1 D2 S **C**: Control word **D1**: 1st destination word **D2**: Filename **S**: 1st source word	Overwrites or appends data in the specified data file in file memory with the specified data from the data area in the CPU Unit. If the specified file doesn't exist, a new file is created with that filename. 	Output	Required
WRITE TEXT FILE	TWRIT @TWRIT	704	TWRIT C S1 S2 S3 S4 **C**: Control word **S1**: Number of bytes to write **S2**: Directory and file name **S3**: Write data **S4**: Delimiter	Reads ASCII data from I/O memory and stores that data in the Memory Card as a text file (writing a new file or appending a file). The data is stored in the TXT format. 	Output	Required

A-1-25 Display Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DISPLAY MESSAGE	MSG @MSG	046	MSG(046) N M **N**: Message number **M**: 1st message word	Reads the specified sixteen words of extended ASCII and displays the message on a Peripheral Device such as a Programming Console.	Output	Required

A-1-26 Clock Instructions

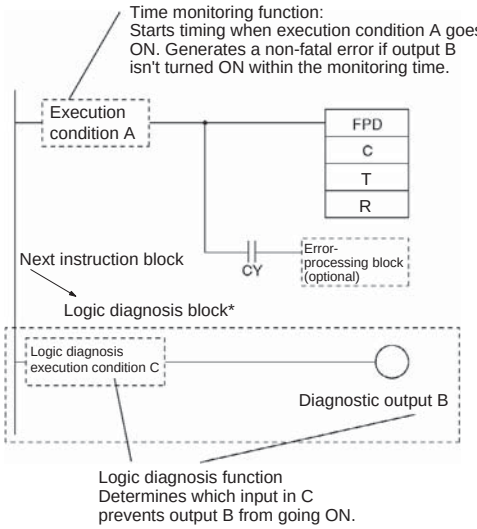
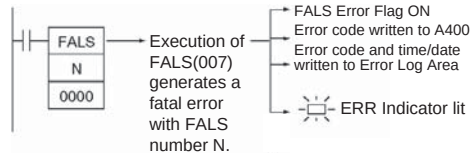
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CALENDAR ADD	CADD @CADD	730	 **C**: 1st calendar word **T**: 1st time word **R**: 1st result word	Adds time to the calendar data in the specified words. 	Output	Required
CALENDAR SUBTRACT	CSUB @CSUB	731	 **C**: 1st calendar word **T**: 1st time word **R**: 1st result word	Subtracts time from the calendar data in the specified words. 	Output	Required
HOURS TO SECONDS	SEC @SEC	065	 **S**: 1st source word **D**: 1st destination word	Converts time data in hours/minutes/seconds format to an equivalent time in seconds only. 	Output	Required
SECONDS TO HOURS	HMS @HMS	066	 **S**: 1st source word **D**: 1st destination word	Converts seconds data to an equivalent time in hours/minutes/seconds format. 	Output	Required

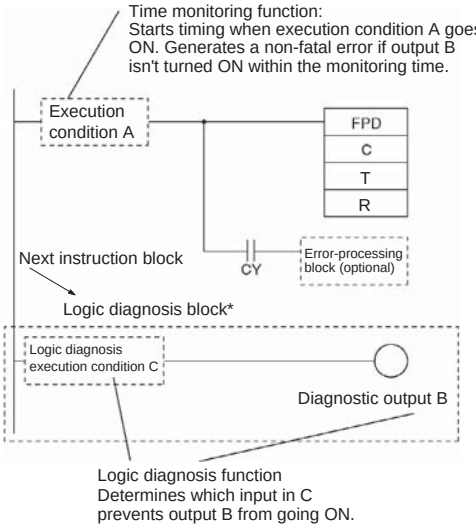
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CLOCK ADJUST-MENT	DATE @DATE	735	DATE(735) S **S**: 1st source word	Changes the internal clock setting to the setting in the specified source words. 	Output	Required

A-1-27 Debugging Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
TRACE MEMORY SAMPLING	TRSM	045	TRSM(045)	When TRSM(045) is executed, the status of a preselected bit or word is sampled and stored in Trace Memory. TRSM(045) can be used anywhere in the program, any number of times.	Output	Not required

A-1-28 Failure Diagnosis Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FAILURE ALARM	FAL @FAL	006	FAL(006) N S **N**: FAL number **S**: 1st message word or error code to generate	Generates or clears user-defined non-fatal errors. Non-fatal errors do not stop PC operation. Also generates non-fatal errors with the system. 	Output	Required
SEVERE FAILURE ALARM	FALS	007	FALS(007) N S **N**: FALS number **S**: 1st message word or error code to generate	Generates user-defined fatal errors. Fatal errors stop PC operation. Also generates fatal errors with the system. 	Output	Required

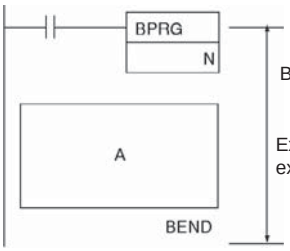
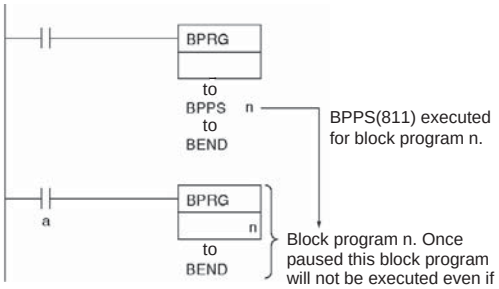
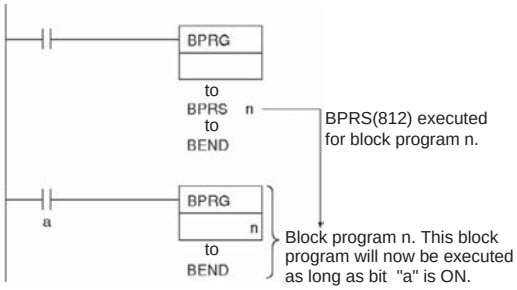
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
FAILURE POINT DETECTION	FPD	269	FPD(269) C T R **C**: Control word **T**: Monitoring time **R**: 1st register word	Diagnoses a failure in an instruction block by monitoring the time between execution of FPD(269) and execution of a diagnostic output and finding which input is preventing an output from being turned ON.  	Output	Required

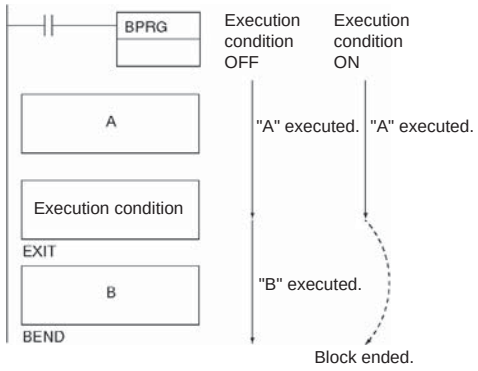
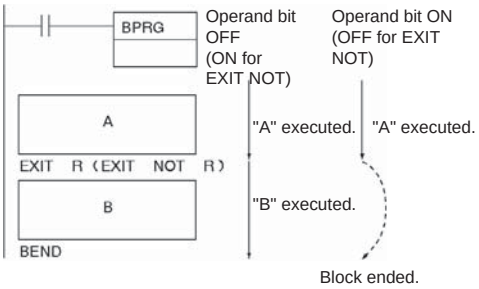
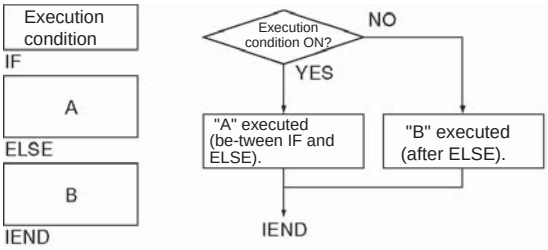
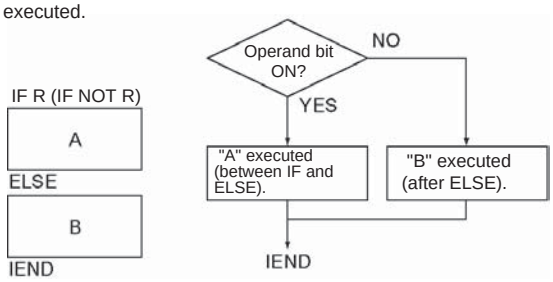
A-1-29 Other Instructions

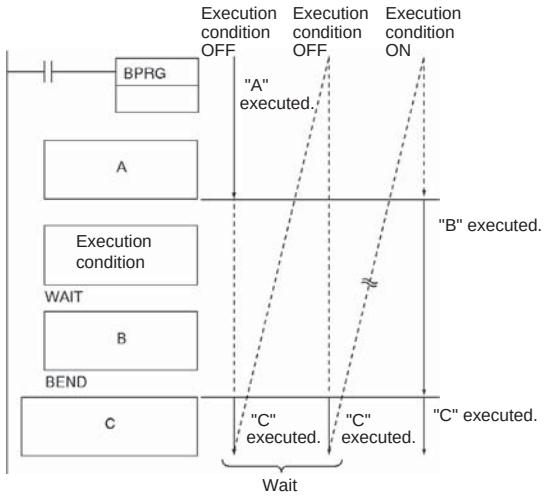
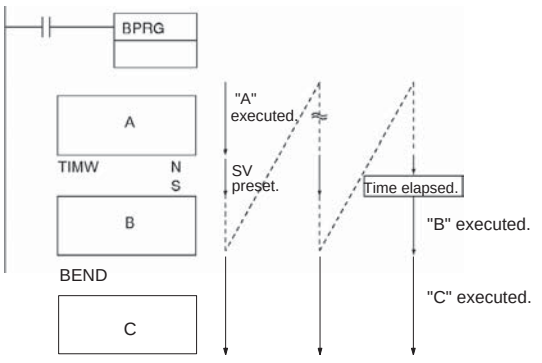
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SET CARRY	STC @STC	040	STC(040)	Sets the Carry Flag (CY).	Output	Required
CLEAR CARRY	CLC @CLC	041	CLC(041)	Turns OFF the Carry Flag (CY).	Output	Required
SELECT EM BANK	EMBC @EMBC	281	EMBC(281) N **N**: EM bank number	Changes the current EM bank.	Output	Required
EXTEND MAXIMUM CYCLE TIME	WDT @WDT	094	WDT(094) T **T**: Timer setting	Extends the maximum cycle time, but only for the cycle in which this instruction is executed.	Output	Required
SAVE CONDITION FLAGS	CCS @CCS	282	CCS(282)	Saves the status of the condition flags.	Output	Required
LOAD CONDITION FLAGS	CCL @CCL	283	CCL(283)	Reads the status of the condition flags that was saved.	Output	Required
CONVERT ADDRESS FROM CV	FRMCV @FRMCV	284	FRMCV(284) S D **S**: Word containing CV-series memory address **D**: Destination Index Register	Converts a CV-series PLC memory address to its equivalent CS/CJ-series PLC memory address.	Output	Required

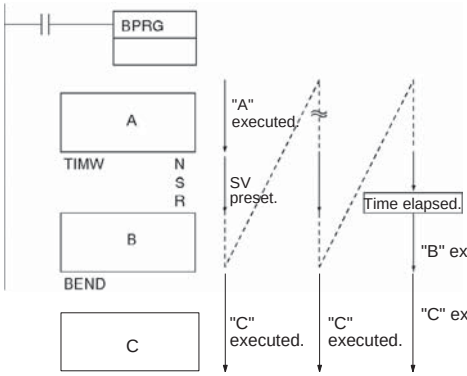
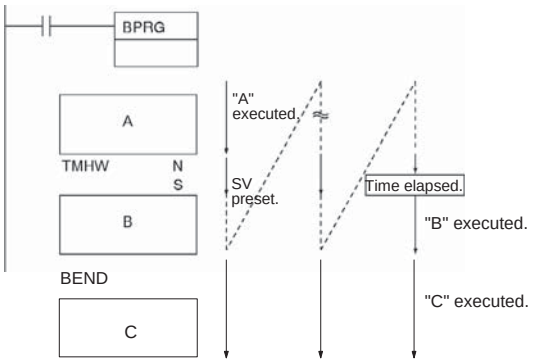
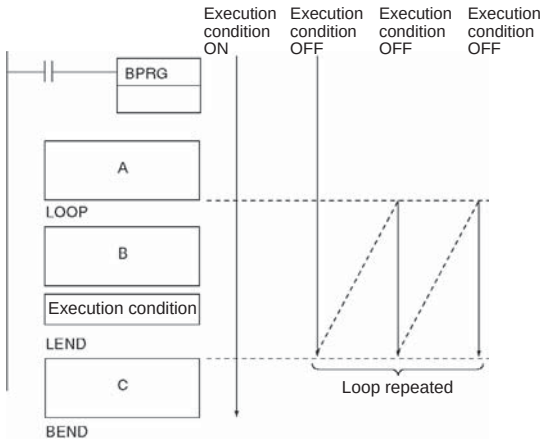
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CONVERT ADDRESS TO CV	TOCV @TOCV	285	TOCV(285) S D **S**: Index Register containing CS-series memory address **D**: Destination word	Converts a CS/CJ-series PLC memory address to its equivalent CVM1/CV-series PLC memory address.	Output	Required

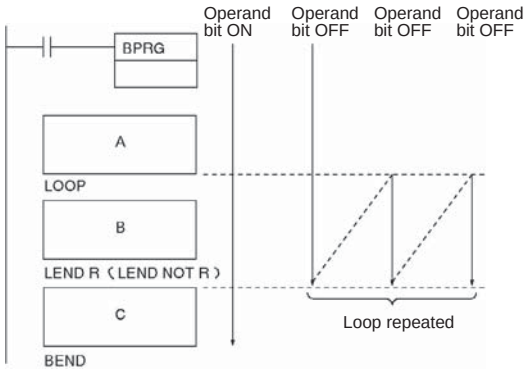
A-1-30 Block Programming Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
BLOCK PROGRAM BEGIN	BPRG	096	BPRG(096) N **N**: Block program number	Define a block programming area. For every BPRG(096) there must be a corresponding BEND(801). 	Output	Required
BLOCK PROGRAM END	BEND	801	---	Define a block programming area. For every BPRG(096) there must be a corresponding BEND(801).	Block program	Required
BLOCK PROGRAM PAUSE	BPPS	811	BPPS (811) N **N**: Block program number	Pause and restart the specified block program from another block program. 	Block program	Required
BLOCK PROGRAM RESTART	BPRS	812	BPRS (812) N **N**: Block program number	Pause and restart the specified block program from another block program. 	Block program	Required

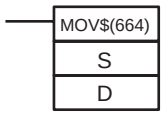
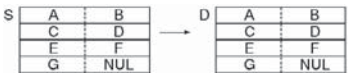
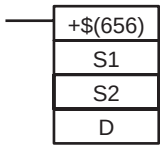
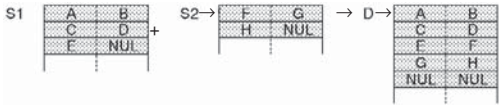
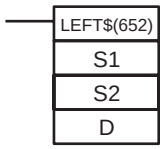
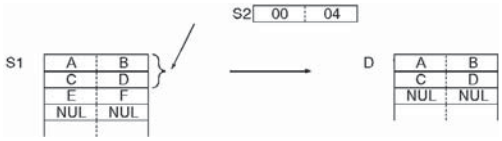
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CONDITIONAL BLOCK EXIT	EXIT	806	EXIT(806) **B**: Bit operand	EXIT(806) without an operand bit exits the program if the execution condition is ON. 	Block program	Required
CONDITIONAL BLOCK EXIT	EXIT	806	EXIT(806)B **B**: Bit operand	EXIT(806) without an operand bit exits the program if the execution condition is ON. 	Block program	Required
CONDITIONAL BLOCK EXIT NOT	EXIT NOT	806	EXIT NOT(806) B **B**: Bit operand	EXIT(806) without an operand bit exits the program if the execution condition is OFF.	Block program	Required
CONDITIONAL BLOCK BRANCHING	IF	802	IF (802)	If the execution condition is ON, the instructions between IF(802) and ELSE(803) will be executed and if the execution condition is OFF, the instructions between ELSE(803) and IEND(804) will be executed. 	Block program	Required
CONDITIONAL BLOCK BRANCHING	IF	802	IF (802) B **B**: Bit operand	If the operand bit is ON, the instructions between IF(802) and ELSE(803) will be executed. If the operand bit is OFF, the instructions between ELSE(803) and IEND(804) will be executed. 	Block program	Required

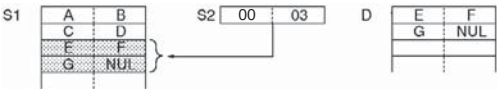
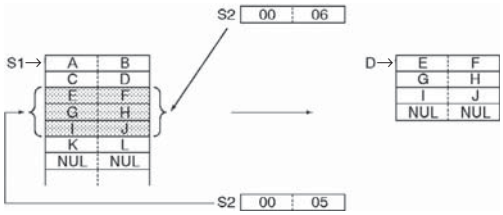
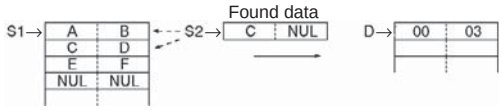
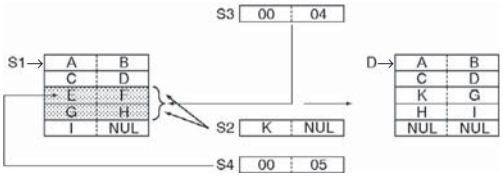
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
CONDITIONAL BLOCK BRANCHING (NOT)	IF NOT	802	IF (802) NOT B **B**: Bit operand	The instructions between IF(802) and ELSE(803) will be executed and if the operand bit is ON, the instructions be ELSE(803) and IEND(804) will be executed is the operand bit is OFF.	Block program	Required
CONDITIONAL BLOCK BRANCHING (ELSE)	ELSE	803	---	If the ELSE(803) instruction is omitted and the operand bit is ON, the instructions between IF(802) and IEND(804) will be executed	Block program	Required
CONDITIONAL BLOCK BRANCHING END	IEND	804	---	If the operand bit is OFF, only the instructions after IEND(804) will be executed.	Block program	Required
ONE CYCLE AND WAIT	WAIT	805	WAIT(805)	If the execution condition is ON for WAIT(805), the rest of the instruction in the block program will be skipped.  	Block program	Required
ONE CYCLE AND WAIT	WAIT	805	WAIT(805) B **B**: Bit operand	If the operand bit is OFF (ON for WAIT NOT(805)), the rest of the instructions in the block program will be skipped. In the next cycle, none of the block program will be executed except for the execution condition for WAIT(805) or WAIT(805) NOT. When the execution condition goes ON (OFF for WAIT(805) NOT), the instruction from WAIT(805) or WAIT(805) NOT to the end of the program will be executed.	Block program	Required
ONE CYCLE AND WAIT (NOT)	WAIT NOT	805	WAIT(805) NOT B **B**: Bit operand	If the operand bit is OFF (ON for WAIT NOT(805)), the rest of the instructions in the block program will be skipped. In the next cycle, none of the block program will be executed except for the execution condition for WAIT(805) or WAIT(805) NOT. When the execution condition goes ON (OFF for WAIT(805) NOT), the instruction from WAIT(805) or WAIT(805) NOT to the end of the program will be executed.	Block program	Required
TIMER WAIT	TIMW (BCD)	813	TIMW(813) N SV **N**: Timer number **SV**: Set value	Delays execution of the block program until the specified time has elapsed. Execution continues from the next instruction after TIMW(813)/TIMWX(816) when the timer times out. SV: 0 to 999.9 s for BCD and 0 to 6,553.5 s for binary  	Block program	Required
	TIMWX (Binary)	816	TIMWX(816) N SV **N**: Timer number **SV:** Set value			

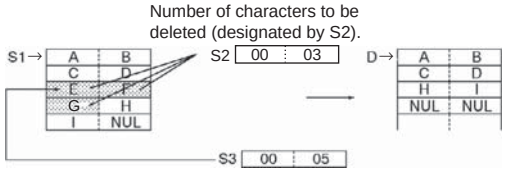
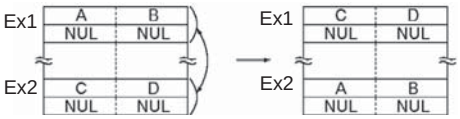
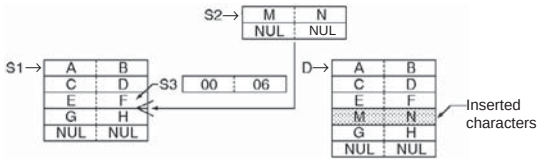
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
COUNTER WAIT	CNTW (BCD)	814	CNTW(814) N SV **N**: Counter number **SV**: Set value **I**: Count input	Delays execution of the rest of the block program until the specified count has been achieved. Execution will be continued from the next instruction after CNTW(814)/CNTWX(817) when the counter counts out. SV: 0 to 9,999 times for BCD and 0 to 65,535 times for binary 	Block program	Required
	CNTWX (Binary)	818	CNTWX(818) N SV **N**: Counter number **SV**: Set value **I**: Count input			
HIGH-SPEED TIMER WAIT	TMHW (BCD)	815	TMHW(815) N SV **N**: Timer number **SV**: Set value	Delays execution of the rest of the block program until the specified time has elapsed. Execution will be continued from the next instruction after TMHW(815) when the timer times out. SV: 0 to 99.99 s for BCD and 0 to 655.35 s for binary 	Block program	Required
	TMHWX (Binary)	817	TMHW(817) N SV **N**: Timer number **SV**: Set value			
LOOP	LOOP	809	---	LOOP(809) designates the beginning of the loop program. 	Block program	Required
LEND	LEND	810	LEND (810)	LEND(810) or LEND(810) NOT specifies the end of the loop. When LEND(810) or LEND(810) NOT is reached, program execution will loop back to the next previous LOOP(809) until the operand bit for LEND(810) or LEND(810) NOT turns ON or OFF (respectively) or until the execution condition for LEND(810) turns ON.	Block program	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
LEND	LEND	810	LEND (810) B **B**: Bit operand	If the operand bit is OFF for LEND(810) (or ON for LEND(810) NOT), execution of the loop is repeated starting with the next instruction after LOOP(809). If the operand bit is ON for LEND(810) (or OFF for LEND(810) NOT), the loop is ended and execution continues to the next instruction after LEND(810) or LEND(810) NOT.  **Note** The status of the operand bit would be reversed for LEND(810) NOT.	Block program	Required
LEND NOT	LEND NOT	810	LEND(810) NOT **B**: Bit operand	LEND(810) or LEND(810) NOT specifies the end of the loop. When LEND(810) or LEND(810) NOT is reached, program execution will loop back to the next previous LOOP(809) until the operand bit for LEND(810) or LEND(810) NOT turns ON or OFF (respectively) or until the execution condition for LEND(810) turns ON.	Block program	Required

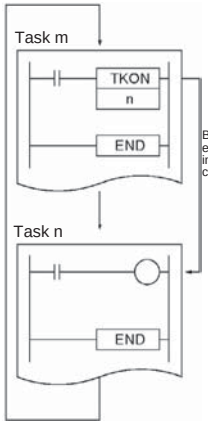
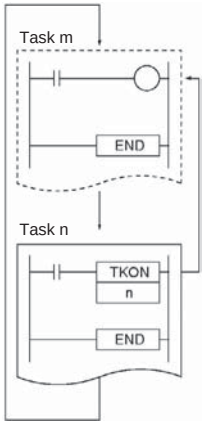
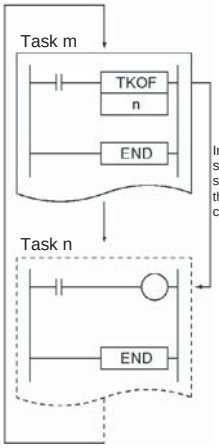
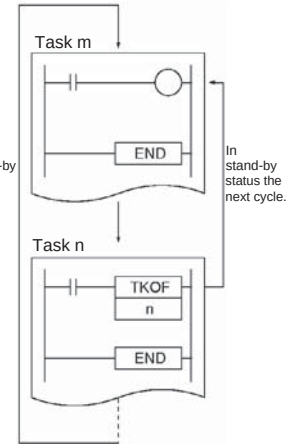
A-1-31 Text String Processing Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
MOV STRING	MOV$ @MOV$	664	 **S**: 1st source word **D**: 1st destination word	Transfers a text string. 	Output	Required
CONCATENATE STRING	+$ @+$	656	 **S1**: Text string 1 **S2**: Text string 2 **D**: First destination word	Links one text string to another text string. 	Output	Required
GET STRING LEFT	LEFT$ @LEFT$	652	 **S1**: Text string first word **S2**: Number of characters **D**: First destination word	Fetches a designated number of characters from the left (beginning) of a text string. 	Output	Required

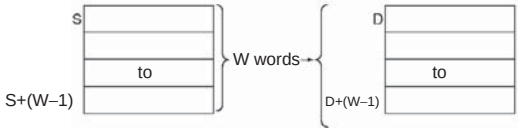
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
GET STRING RIGHT	RGHT$ @RGHT$	653	RGHT$(653) S1 S2 D **S1**: Text string first word **S2**: Number of characters **D**: First destination word	Reads a designated number of characters from the right (end) of a text string. 	Output	Required
GET STRING MIDDLE	MID$ @MID$	654	MID$(654) S1 S2 S3 D **S1**: Text string first word **S2**: Number of characters **S3**: Beginning position **D**: First destination word	Reads a designated number of characters from any position in the middle of a text string. 	Output	Required
FIND IN STRING	FIND @FIND$	660	FIND$(660) S1 S2 D **S1**: Source text string first word **S2**: Found text string first word **D**: First destination word	Finds a designated text string from within a text string. 	Output	Required
STRING LENGTH	LEN$ @LEN$	650	LEN$(650) S D **S**: Text string first word **D**: 1st destination word	Calculates the length of a text string. 	Output	Required
REPLACE IN STRING	RPLC$ @RPLC$	661	RPLC$(654) S1 S2 S3 S4 D **S1**: Text string first word **S2**: Replacement text string first word **S3**: Number of characters **S4**: Beginning position **D**: First destination word	Replaces a text string with a designated text string from a designated position. 	Output	Required

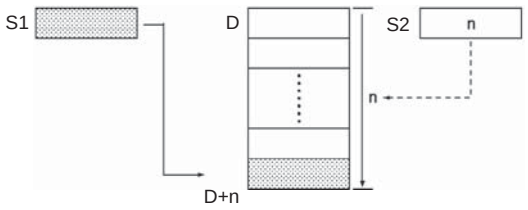
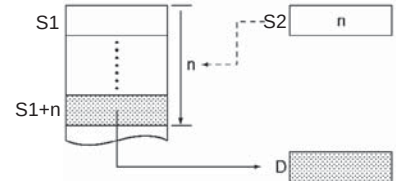
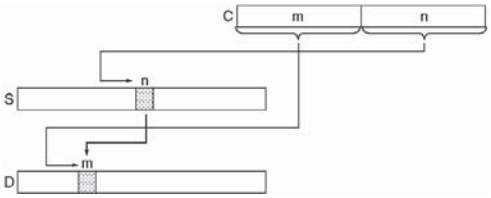
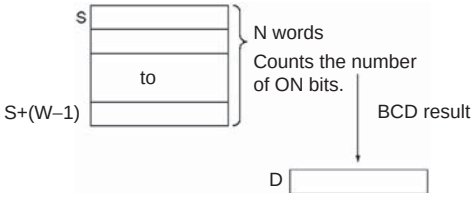
Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
DELETE STRING	DEL$ @DEL$	658	DEL$(658) S1 S2 S3 D **S1**: Text string first word **S2**: Number of characters **S3**: Beginning position **D**: First destination word	Deletes a designated text string from the middle of a text string. Number of characters to be deleted (designated by S2). 	Output	Required
EXCHANGE STRING	XCHG$ @XCHG$	665	XCHG$(665) Ex1 Ex2 **Ex1**: 1st exchange word 1 **Ex2**: 1st exchange word 2	Replaces a designated text string with another designated text string. 	Output	Required
CLEAR STRING	CLR$ @CLR$	666	CLR$(666) S **S**: Text string first word	Clears an entire text string with NUL (00 hex). 	Output	Required
INSERT INTO STRING	INS$ @INS$	667	INS$(657) S1 S2 S3 D **S1**: Base text string first word **S2**: Inserted text string first word **S3**: Beginning position **D**: First destination word	Deletes a designated text string from the middle of a text string. 	Output	Required
String Comparison	LD, AND, OR + =$, <>$, <$, <=$, >$, >=$	670 (=$) 671 (<>$) 672 (<$) 673 (<=$) 674 (>$) 675 (>=$)	Symbol S1 S2 Symbol S1 S2 Symbol S1 S2 **S1**: Text string 1 **S2**: Text string 2	Sting comparison instructions (=$, <>$, <$, <=$, >$, >=$) compare two text strings from the beginning, in terms of value of the ASCII codes. If the result of the comparison is true, an ON execution condition is created for a LOAD, AND, or OR.	LD: Logic start AND, OR: Continues on rung	Required

A-1-32 Task Control Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
TASK ON	TKON @TKON	820	TKON(820) N **N**: Task number	Makes the specified task executable. The specified task's task number is higher than the local task's task number (m<n).  The specified task's task number is lower than the local task's task number (m>n). 	Output	Required
TASK OFF	TKOF @TKOF	821	TKOF(821) N **N**: Task number	Puts the specified task into standby status. The specified task's task number is higher than the local task's task number (m<n).  The specified task's task number is lower than the local task's task number (m>n). 	Output	Required

A-1-33 Model Conversion Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
BLOCK TRANSFER	XFERC @XFERC	565	XFERC(565) N S D **N**: Number of words **S**: 1st source word **D**: 1st destination word	Transfers the specified number of consecutive words. 	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
SINGLE WORD DISTRIBUTE	DISTC @DISTC	566	DISTC(566) S Bs Of **S**: Source word **Bs**: Destination base address **Of**: Offset	Transfers the source word to a destination word calculated by adding an offset value to the base address.  Can also write to a stack (Stack Push Operation).	Output	Required
DATA COLLECT	COLLC @COLLC	567	COLLC(567) Bs Of D **Bs**: Source base address **Of**: Offset (BCD) **D**: Destination word	Transfers the source word (calculated by adding an offset value to the base address) to the destination word.  Can also read data from a stack (Stack Read Operation).	Output	Required
MOVE BIT	MOVBC @MOVBC	568	MOVBC(568) S C D **S**: Source word or data **C**: Control word (BCD) **D**: Destination word	Transfers the specified bit. 	Output	Required
BIT COUNTER	BCNTC @BCNTC	621	BCNTC(621) N S R **N**: Number of words (BCD) **S**: 1st source word **R**: Result word	Counts the total number of ON bits in the specified word(s). 	Output	Required

A-1-34 Special Function Block Instructions

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
GET VARIABLE ID	GETID @GETID	286	GETID(286) S D1 D2 **S**: Variable or address **D1**: ID code **D2**: Destination word	Outputs the FINS command variable type (data area) code and word address for the specified variable or address. This instruction is generally used to get the assigned address of a variable in a function block.	Output	Required
STEP ACTIVATE	SA @SA	784	SA(784) D	Makes the specified step or subchart active to start execution of the actions.	Output	Required

Instruction	Mnemonic	Code	Symbol/Operand	Function	Location	Execution condition
STEP DEACTIVATE	SE @SE	785	SE(784) D	Makes the specified step of subchart inactive to end execution of the actions.	Output	Required
READ SET TIMER	TSR @TSR	780	TSR(780) S D	The present value of the Step Time specified by S is stored starting at D.	Output	Required
SET STEP TIMER	TSW @TSW	781	TSW(781) S D	The present value of the Step Timer specified by S is changed to the value specified starting at D.	Output	Required
SFC ON	SFCON	789	SFCON(789) N **N**: SFC task number	Restarts execution of an SFC task that was ended or paused using one of the other SFC Task Control Instructions.	Output	Required
SFC OFF	SFCOFF	790	SFCOFF(790) N **N**: SFC task number	Ends execution of an SFC task. The status of all outputs is held. When execution of the SFC task is restarted, it is executed from the initial step.	Output	Required
SFC PAUSE WITH RESET	SFCPR	793	SFCPR(793) N **N**: SFC task number	Pauses execution of an SFC task. The status of all outputs is reset.	Output	Required
SFC PAUSE WITH NO RESET	SFCPRN	791	SFCPRN(791) N **N**: SFC task number	Pauses execution of an SFC task. The status of all outputs is held.	Output	Required

A-2 Instruction Execution Times and Number of Steps

The following table lists the execution times for all instructions that are supported by the CPU Units.

The total execution time of instructions within one whole user program is the process time for program execution when calculating the cycle time (*1).

*1 User programs are allocated tasks that can be executed within cyclic tasks and interrupt tasks that satisfy interrupt conditions.

Execution times for most instructions differ depending on the CPU Unit used and the conditions when the instruction is executed.

The execution time can also vary when the execution condition is OFF.

The following table also lists the length of each instruction in the *Length (steps)* column. The number of steps required in the user program area for each instructions depends on the instruction and the operands used with it.

The number of steps in a program is not the same as the number of instructions.

Note 1 Most instructions are supported in differentiated form (indicated with ↑, ↓, @, and %). Specifying differentiation will increase the execution times by the following amounts.

(unit: μs)

Symbol	CJ2 CPU Unit	
	CJ2H-CPU6□(-EIP)	CJ2M-CPU□□
↑ or ↓	+0.24	+0.32
@ or %	+0.24	+0.32

2 Use the following time as a guideline when instructions are not executed.

(unit: μs)

CJ2 CPU Unit	
CJ2H-CPU6□(-EIP)	CJ2M-CPU□□
0.016	0.020

Execution times will vary somewhat depending on the order in which instructions are executed and on whether instructions are executed. Use the following formula to calculate the maximum variation for a worst-case scenario.

CJ2H CPU Units: Number of instruction steps × 0.016 (μs)
CJ2M CPU Units: Number of instruction steps × 0.020 (μs)

A-2-1 Sequence Input Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
LOAD	LD	---	1 to 2	0.016	0.040	---
	!LD	---	2 to 14	0.99	1.260	---
LOAD NOT	LD NOT	---	1 to 2	0.016	0.040	---
	!LD NOT	---	2 to 14	0.99	1.260	---
AND	AND	---	1 to 2	0.016	0.040	---
	!AND	---	2 to 14	0.99	1.260	---
AND NOT	AND NOT	---	1 to 2	0.016	0.040	---
	!AND NOT	---	2 to 14	0.99	1.260	---
OR	OR	---	1 to 2	0.016	0.040	---
	!OR	---	2 to 14	0.99	1.260	---
OR NOT	OR NOT	---	1 to 2	0.016	0.040	---
	!OR NOT	---	2 to 14	0.99	1.260	---
AND LOAD	AND LD	---	1	0.016	0.040	---
OR LOAD	OR LD	---	1	0.016	0.040	---
NOT	NOT	520	1	0.016	0.040	---
CONDITION ON	UP	521	3	0.26	0.36	---
CONDITION OFF	DOWN	522	4	0.27	0.40	---
LOAD BIT TEST	LD TST	350	4	0.11	0.16	---
LOAD BIT TEST NOT	LD TSTN	351	4	0.11	0.16	---
AND BIT TEST	AND TST	350	4	0.11	0.16	
AND BIT TEST NOT	AND TSTN	351	4	0.11	0.16	---
OR BIT TEST	OR TST	350	4	0.11	0.16	---
OR BIT TEST NOT	OR TSTN	351	4	0.11	0.16	---

A-2-2 Sequence Output Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
OUTPUT	OUT	---	1 to 2	0.016	0.040	---
	!OUT	---	2 to 14	0.99	1.320	---
OUTPUT NOT	OUT NOT	---	1 to 2	0.016	0.040	---
	!OUT NOT	---	2 to 14	0.99	1.320	---
KEEP	KEEP	011	1 to 2	0.048	0.060	---
	!KEEP	011	16	0.99	1.340	---
DIFFERENTIATE UP	DIFU	013	2 to 2	0.28	0.30	---
DIFFERENTIATE DOWN	DIFD	014	2 to 2	0.24	0.30	---
SET	SET	---	1 to 2	0.016	0.040	---
	!SET	---	2 to 14	0.99	1.360	---
RESET	RSET	---	1 to 2	0.016	0.040	---
	!RSET	---	2 to 14	0.99	1.360	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
MULTIPLE BIT SET	SETA	530	4	3.68	4.12	With 1-bit set
				15.5	24.4	With 1,000-bit set
MULTIPLE BIT RESET	RSTA	531	4	3.7	4.1	With 1-bit reset
				15.5	24.4	With 1,000-bit reset
SINGLE BIT SET	SETB	532	2	0.19	0.280	---
	!SETB	---	16	0.99	1.120	---
SINGLE BIT RESET	RSTB	534	2	0.19	0.280	---
	!RSTB	---	16	0.99	1.120	---
SINGLE BIT OUTPUT	OUTB	534	2	0.19	0.280	---
	!OUTB	---	16	0.99	1.180	---

A-2-3 Sequence Control Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
END	END	001	1	2.6	3.5	---
NO OPERATION	NOP	000	1	0.016	0.040	---
INTERLOCK	IL	002	1	0.048	0.060	---
INTERLOCK CLEAR	ILC	003	1	0.048	0.060	---
MULTI-INTERLOCK DIFFERENTIATION HOLD	MILH	517	3	2.3	3.3	Interlock condition not met (input condition ON)
				3.4	4.6	Interlock condition met (input condition OFF)
				3.8	5.2	Interlock condition met again during interlock (input condition OFF)
MULTI-INTERLOCK DIFFERENTIATION RELEASE	MILR	518	3	2.3	3.1	Interlock condition not met (input condition ON)
				3.4	4.5	Interlock condition met (input condition OFF)
				3.8	5.1	Interlock condition met again during interlock (input condition OFF)
MULTI-INTERLOCK CLEAR	MILC	519	2	1.2	1.7	Not during interlock
				1.6	2.2	During interlock
JUMP	JMP	004	2	0.31	0.34	---
JUMP END	JME	005	2	---	---	---
CONDITIONAL JUMP	CJP	510	2	0.31	0.34	Jump condition met (input condition ON)

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
CONDITIONAL JUMP NOT	CJPN	511	2	0.31	0.34	Jump condition met (input condition OFF)
MULTIPLE JUMP	JMP0	515	1	0.048	0.060	---
MULTIPLE JUMP END	JME0	516	1	0.048	0.060	---
FOR LOOP	FOR	512	2	0.27	0.42	Designating a constant
BREAK LOOP	BREAK	514	1	0.048	0.060	---
NEXT LOOP	NEXT	513	1	0.14	0.16	When loop is continued
				0.18	0.18	When loop is ended

A-2-4 Timer and Counter Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
HUNDRED-MS TIMER	TIM	---	3	0.67	0.84	---
	TIMX	550		0.67	0.84	
TEN-MS TIMER	TIMH	015	3	0.67	0.84	---
	TIMHX	551		0.67	0.84	
ONE-MS TIMER	TMHH	540	3	0.67	0.84	---
	TMHHX	552		0.67	0.84	
TENTH-MS TIMER	TIMU	541	3	0.67	0.84	---
	TIMUX	556		0.67	0.84	
HUNDREDTH-MS TIMER	TMUH	544	3	0.67	0.84	---
	TMUHX	557		0.67	0.84	
ACCUMULATIVE TIMER	TTIM	087	3	9.2	12.1	---
				6.9	8.4	When resetting
				5.0	6.5	When interlocking
	TTIMX	555	3	8.8	11.7	---
				6.8	8.5	When resetting
				5.0	6.5	When interlocking
LONG TIMER	TIML	542	4 to 5	5.8	7.0	---
				3.9	4.1	When interlocking
	TIMLX	553	4 to 5	5.7	7.0	---
				3.6	3.7	When interlocking
MULTI-OUTPUT TIMER	MTIM	543	4	6.4	7.2	---
				3.7	4.3	When resetting
	MTIMX	554	4	5.5	6.4	---
				3.4	3.8	When resetting
TIMER RESET	TRSET	549	2	0.58	0.8	---
COUNTER	CNT	---	3	0.51	0.58	---
	CNTX	546		0.51	0.58	
REVERSIBLE COUNTER	CNTR	012	3	9.1	11.8	---
	CNTRX	548		8.0	10.3	

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
RESET TIMER/COUNTER	CNR	545	3	4.8	5.4	When resetting 1 word
				2839	2555	When resetting 1,000 words
	CNRX	547	3	4.7	5.5	When resetting 1 word
				2839	2555	When resetting 1,000 words

A-2-5 Comparison Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
Input Comparison Instructions (unsigned)	=	300	4	0.08	0.16	---
	<>	305				
	<	310				
	<=	315				
	>	320				
	>=	325				
Input Comparison Instructions (double, unsigned)	=L	301	4 to 7	0.08	0.24	---
	<>L	306				
	<L	311				
	<=L	316				
	>L	321				
	>=L	326				
Input Comparison Instructions (signed)	=S	302	4	0.08	0.16	---
	<>S	307				
	<S	312				
	<=S	317				
	>S	322				
	>=S	327				
Input Comparison Instructions (double, signed)	=SL	303	4 to 7	0.08	0.24	---
	<>SL	308				
	<SL	313				
	<=SL	318				
	>SL	323				
	>=SL	328				
Time Comparison Instructions	=DT	341	4	16.300	27.9	---
	<>DT	342				
	<DT	343				
	<=DT	344				
	>DT	345				
	>=DT	346				
COMPARE	CMP	020	3	0.06	0.080	---
	!CMP	020	30	2.06	2.6	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DOUBLE COMPARE	CMPL	060	3 to 5	0.064	0.120	---
SIGNED BINARY COMPARE	CPS	114	3 to 5	0.064	0.080	---
	!CPS	114	30	2.06	2.6	---
DOUBLE SIGNED BINARY COMPARE	CPSL	115	3 to 5	0.064	0.120	---
TABLE COMPARE	TCMP	085	4	10.3	12.5	---
MULTIPLE COMPARE	MCMP	019	4	15.2	20.3	---
UNSIGNED BLOCK COMPARE	BCMP	068	4	16.3	20.5	---
EXPANDED BLOCK COMPARE	BCMP2	502	4	5.0	5.1	Number of data words: 1
				217.2	278	Number of data words: 255
AREA RANGE COMPARE	ZCP	088	3	0.14	0.400	---
DOUBLE AREA RANGE COMPARE	ZCPL	116	3 to 5	0.14	0.640	---
SIGNED AREA RANGE COMPARE	ZCPS	117	3	0.14	0.400	---
DOUBLE SIGNED AREA RANGE COMPARE	ZCPSL	118	3 to 5	0.14	0.640	---

A-2-6 Data Movement Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
MOVE	MOV	021	3	0.05	0.12	---
	!MOV	021	30	1.98	2.6	---
DOUBLE MOVE	MOVL	498	3 to 4	0.05	0.20	---
MOVE NOT	MVN	022	3	0.05	0.12	---
DOUBLE MOVE NOT	MVNL	499	3 to 4	0.05	0.20	---
MOVE BIT	MOVB	082	4	0.19	0.32	---
MOVE DIGIT	MOVD	083	4	0.19	0.32	---
MULTIPLE BIT TRANSFER	XFRB	062	4	6.6	9.4	Transferring 1 bit
				85.8	119	Transferring 255 bits
BLOCK TRANSFER	XFER	070	4	0.29	0.28	Transferring 1 word
				240.1	220	Transferring 1,000 words
BLOCK SET	BSET	071	4	0.21	0.20	Setting 1 word
				142.2	140	Setting 1,000 words
DATA EXCHANGE	XCHG	073	3	0.32	0.48	---
DOUBLE DATA EXCHANGE	XCGL	562	3 to 4	0.12	0.29	---
SINGLE WORD DISTRIBUTE	DIST	080	4	4.5	4.7	---
DATA COLLECT	COLL	081	4	4.6	4.7	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
MOVE TO REGISTER	MOVR	560	3	0.064	0.200	---
MOVE TIMER/COUNTER PV TO REGISTER	MOVRW	561	3	0.064	0.200	---

A-2-7 Data Shift Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SHIFT REGISTER	SFT	010	3	2.86	3.47	Shifting 1 word
				315	422	Shifting 1,000 words
REVERSIBLE SHIFT REGISTER	SFTR	084	4	6.22	6.38	Shifting 1 word
				319	422	Shifting 1,000 words
ASYNCHRONOUS SHIFT REGISTER	ASFT	017	4	5.3	6.3	Shifting 1 word
				948	1285	Shifting 1,000 words*1
WORD SHIFT	WSFT	016	4	2.3	3.1	Shifting 1 word
				233	187	Shifting 1,000 words
ARITHMETIC SHIFT LEFT	ASL	025	2	0.18	0.260	---
DOUBLE SHIFT LEFT	ASLL	570	2	0.32	0.420	---
ARITHMETIC SHIFT RIGHT	ASR	026	2	0.18	0.260	---
DOUBLE SHIFT RIGHT	ASRL	571	2	0.32	0.420	---
ROTATE LEFT	ROL	027	2	0.18	0.260	---
DOUBLE ROTATE LEFT	ROLL	572	2	0.32	0.420	---
ROTATE LEFT WITH-OUT CARRY	RLNC	574	2	0.18	0.260	---
DOUBLE ROTATE LEFT WITHOUT CARRY	RLNL	576	2	0.32	0.420	---
ROTATE RIGHT	ROR	028	2	0.18	0.260	---
DOUBLE ROTATE RIGHT	RORL	573	2	0.32	0.420	---
ROTATE RIGHT WITHOUT CARRY	RRNC	575	2	0.18	0.260	---
DOUBLE ROTATE RIGHT WITHOUT CARRY	RRNL	577	2	0.32	0.420	---
ONE DIGIT SHIFT LEFT	SLD	074	3	3.7	4.4	Shifting 1 word
				317.8	429	Shifting 1,000 words
ONE DIGIT SHIFT RIGHT	SRD	075	3	4.5	5.4	Shifting 1 word
				479.5	656	Shifting 1,000 words
SHIFT N-BIT DATA LEFT	NSFL	578	4	4.6	5.2	Shifting 1 bit
				31.5	36.1	Shifting 1,000 bits

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SHIFT N-BIT DATA RIGHT	NSFR	579	4	4.5	5.2	Shifting 1 bit
				39.0	50.2	Shifting 1,000 bits
SHIFT N-BITS LEFT	NASL	580	3	0.18	0.38	---
DOUBLE SHIFT N-BITS LEFT	NSLL	582	3	0.32	0.54	---
SHIFT N-BITS RIGHT	NASR	581	3	0.18	0.38	---
DOUBLE SHIFT N-BITS RIGHT	NSRL	583	3	0.32	0.54	---

*1 The instruction execution time is greatly affected by the amount to data. This will affect the cycle time. To reduce the effect on the cycle time, background execution can be specified. Refer to *10-2-5 Background Execution* for details.

A-2-8 Increment/Decrement Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
INCREMENT BINARY	++	590	2	0.18	0.24	---
DOUBLE INCREMENT BINARY	++L	591	2	0.18	0.24	---
DECREMENT BINARY	– –	592	2	0.18	0.24	---
DOUBLE DECREMENT BINARY	– –L	593	2	0.18	0.24	---
INCREMENT BCD	++B	594	2	3.0	3.4	---
DOUBLE INCREMENT BCD	++BL	595	2	3.2	3.5	---
DECREMENT BCD	– –B	596	2	3.0	3.5	---
DOUBLE DECREMENT BCD	– –BL	597	2	3.2	3.5	---

A-2-9 Symbol Math Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SIGNED BINARY ADD WITHOUT CARRY	+	400	4	0.18	0.34	---
DOUBLE SIGNED BINARY ADD WITHOUT CARRY	+L	401	4 to 6	0.18	0.24	---
SIGNED BINARY ADD WITH CARRY	+C	402	4	0.18	0.34	---
DOUBLE SIGNED BINARY ADD WITH CARRY	+CL	403	4 to 6	0.18	0.24	---
BCD ADD WITHOUT CARRY	+B	404	4	4.0	4.8	---
DOUBLE BCD ADD WITHOUT CARRY	+BL	405	4 to 6	4.9	6.0	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
BCD ADD WITH CARRY	+BC	406	4	4.4	5.2	---
DOUBLE BCD ADD WITH CARRY	+BCL	407	4 to 6	5.2	6.6	---
SIGNED BINARY SUBTRACT WITHOUT CARRY	–	410	4	0.18	0.340	---
DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY	–L	411	4 to 6	0.18	0.24	---
SIGNED BINARY SUBTRACT WITH CARRY	–C	412	4	0.18	0.340	---
DOUBLE SIGNED BINARY SUBTRACT WITH CARRY	–CL	413	4 to 6	0.18	0.24	---
BCD SUBTRACT WITHOUT CARRY	–B	414	4	4.1	4.9	---
DOUBLE BCD SUBTRACT WITHOUT CARRY	–BL	415	4 to 6	4.9	5.9	---
BCD SUBTRACT WITH CARRY	–BC	416	4	4.5	5.2	---
DOUBLE BCD SUBTRACT WITH CARRY	–BCL	417	4 to 6	5.2	6.3	---
SIGNED BINARY MULTIPLY	*	420	4	0.26	0.520	---
DOUBLE SIGNED BINARY MULTIPLY	*L	421	4 to 6	3.6	3.9	---
UNSIGNED BINARY MULTIPLY	*U	422	4	0.26	0.26	---
DOUBLE UNSIGNED BINARY MULTIPLY	*UL	423	4 to 6	3.6	3.9	---
BCD MULTIPLY	*B	424	4	3.6	4.6	---
DOUBLE BCD MULTIPLY	*BL	425	4 to 6	4.9	6.2	---
SIGNED BINARY DIVIDE	/	430	4	0.29	0.540	---
DOUBLE SIGNED BINARY DIVIDE	/L	431	4 to 6	4.2	4.8	---
UNSIGNED BINARY DIVIDE	/U	432	4	0.29	0.540	---
DOUBLE UNSIGNED BINARY DIVIDE	/UL	433	4 to 6	3.8	4.2	---
BCD DIVIDE	/B	434	4	5.0	5.9	---
DOUBLE BCD DIVIDE	/BL	435	4 to 6	4.8	5.9	---

A-2-10 Conversion Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
BCD TO BINARY	BIN	023	3	0.18	0.280	---
DOUBLE BCD TO DOUBLE BINARY	BINL	058	3 to 4	3.3	3.5	---
BINARY TO BCD	BCD	024	3	0.19	0.300	---
DOUBLE BINARY TO DOUBLE BCD	BCDL	059	3 to 4	3.3	3.7	---
2'S COMPLEMENT	NEG	160	3	0.14	0.240	---
DOUBLE 2'S COMPLEMENT	NEGL	161	3 to 4	0.26	0.440	---
16-BIT TO 32-BIT SIGNED BINARY	SIGN	600	3	0.26	0.340	---
DATA DECODER	MLPX	076	4	0.17	0.280	Decoding 1 digit (4 to 16)
				0.42	0.770	Decoding 4 digits (4 to 16)
				1.14	1.760	Decoding 1 digit 8 to 256
				2.17	3.370	Decoding 4 digits (8 to 256)
DATA ENCODER	DMPX	077	4	3.3	4.6	Encoding 1 digit (16 to 4)
				3.7	5.2	Encoding 4 digits (16 to 4)
				17.3	26.3	Encoding 1 digit (256 to 8)
				35	47	Encoding 2 digits (256 to 8)
ASCII CONVERT	ASC	086	4	4.0	4.5	Converting 1 digit into ASCII
				4.6	5.2	Converting 4 digits into ASCII
ASCII TO HEX	HEX	162	4	3.3	3.8	Converting 1 digit
COLUMN TO LINE	LINE	063	4	10.5	13.1	---
LINE TO COLUMN	COLM	064	4	13.8	17.6	---
SIGNED BCD TO BINARY	BINS	470	4	3.6	4.0	Data format setting No. 0
				3.6	4.0	Data format setting No. 1
				3.6	4.0	Data format setting No. 2
				3.6	4.0	Data format setting No. 3

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DOUBLE SIGNED BCD TO BINARY	BISL	472	4 to 5	3.7	4.1	Data format setting No. 0
				3.6	4.1	Data format setting No. 1
				3.7	4.2	Data format setting No. 2
				3.7	4.2	Data format setting No. 3
SIGNED BINARY TO BCD	BCDS	471	4	3.7	4.0	Data format setting No. 0
				3.7	4.1	Data format setting No. 1
				3.7	4.2	Data format setting No. 2
				3.7	4.2	Data format setting No. 3
DOUBLE SIGNED BINARY TO BCD	BDSL	473	4 to 5	4.0	4.5	Data format setting No. 0
				4.0	4.6	Data format setting No. 1
				4.0	4.6	Data format setting No. 2
				4.1	4.6	Data format setting No. 3
GRAY CODE CONVERSION	GRY	474	4	26.5	49.1	8-bit binary
				27.6	51.1	8-bit BCD
				30.9	57.2	8-bit angle
				35.3	66.0	15-bit binary
				36.3	68.0	15-bit BCD
				39.6	74.0	15-bit angle
GRAY CODE TO BINARY CONVERT	GRAY_BIN	478	3	0.1	0.3	---
DOUBLE GRAY CODE TO BINARY CONVERT	GRAY_BINL	479	3 to 4	0.1	0.4	---
BINARY TO GRAY CODE CONVERT	BIN_GRAY	480	3	0.1	0.3	---
DOUBLE BINARY TO GRAY CODE CONVERT	BIN_GRAYL	481	3 to 4	0.1	0.4	---
FOUR-DIGIT NUMBER TO ASCII	STR4	601	3	8.4	14.2	---
EIGHT-DIGIT NUMBER TO ASCII	STR8	602	3 to 4	10.2	16.4	---
SIXTEEN-DIGIT NUMBER TO ASCII	STR16	603	3	15.8	28.2	---
ASCII TO FOUR-DIGIT NUMBER	NUM4	604	3 to 4	10.5	18.5	---
ASCII TO EIGHT-DIGIT NUMBER	NUM8	605	3	14.8	27.1	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
ASCII TO SIXTEEN-DIGIT NUMBER	NUM16	606	3	27.4	52.0	---

A-2-11 Logic Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
LOGICAL AND	ANDW	034	4	0.14	0.340	---
DOUBLE LOGICAL AND	ANDL	610	4 to 6	0.26	0.640	---
LOGICAL OR	ORW	035	4	0.18	0.340	---
DOUBLE LOGICAL OR	ORWL	611	4 to 6	0.26	0.640	---
EXCLUSIVE OR	XORW	036	4	0.18	0.340	---
DOUBLE EXCLUSIVE OR	XORL	612	4 to 6	0.26	0.640	---
EXCLUSIVE NOR	XNRW	037	4	0.18	0.340	---
DOUBLE EXCLUSIVE NOR	XNRL	613	4 to 6	0.26	0.640	---
COMPLEMENT	COM	029	2	0.18	0.240	---
DOUBLE COMPLEMENT	COML	614	2	0.32	0.440	---

A-2-12 Special Math Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
BINARY ROOT	ROTB	620	3	15.4	24.2	---
BCD SQUARE ROOT	ROOT	072	3	17.1	25.3	---
ARITHMETIC PROCESS	APR	069	4	4.6	5.3	Designating SIN and COS
				5.7	6.9	Designating line-segment approximation
FLOATING POINT DIVIDE	FDIV	079	4	76	149	---
BIT COUNTER	BCNT	067	4	0.24	0.360	Counting 1 word

A-2-13 Floating-point Math Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
FLOATING TO 16-BIT	FIX	450	3 to 4	0.13	0.24	---
FLOATING TO 32-BIT	FIXL	451	3 to 4	0.13	0.32	---
16-BIT TO FLOATING	FLT	452	3 to 4	0.13	0.30	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
32-BIT TO FLOATING	FLTL	453	3 to 4	0.13	0.32	---
FLOATING-POINT ADD	+F	454	4 to 6	0.24	0.66	---
FLOATING-POINT SUBTRACT	–F	455	4 to 6	0.24	0.66	---
FLOATING-POINT DIVIDE	/F	457	4 to 6	0.4	0.9	---
FLOATING-POINT MULTIPLY	*F	456	4 to 6	0.24	0.66	---
DEGREES TO RADIANS	RAD	458	3 to 4	2.7	3.3	---
RADIANS TO DEGREES	DEG	459	3 to 4	3.0	3.2	---
SINE	SIN	460	3 to 4	3.8	4.3	0° specified
				4.5	5.4	45° specified
				5.0	6.0	90° specified
HIGH-SPEED SINE	SINQ	475	8 to 9	0.59	0.86	0°, 45°, or 90° specified
COSINE	COS	461	3 to 4	3.7	4.3	0° specified
				4.4	5.2	45° specified
				5.3	6.7	90° specified
HIGH-SPEED COSINE	COSQ	476	8 to 9	0.59	0.86	0°, 45°, or 90° specified
TANGENT	TAN	462	3 to 4	3.9	4.5	0° specified
				6.1	8.2	45° specified
HIGH-SPEED TANGENT	TANQ	477	15 to 16	1.2	1.7	0°, 45°, or 90° specified
ARC SINE	ASIN	463	3 to 4	5.8	7.1	0° specified
				24.8	33.0	45° specified
				5.6	7.0	90° specified
ARC COSINE	ACOS	464	3 to 4	5.3	6.8	0° specified
				27.2	34.6	45° specified
				6.4	7.1	90° specified
ARC TANGENT	ATAN	465	3 to 4	4.0	5.0	0° specified
				5.6	7.0	45° specified
SQUARE ROOT	SQRT	466	3 to 4	0.42	0.66	---
EXPONENT	EXP	467	3 to 4	3.8	4.5	---
LOGARITHM	LOG	468	3 to 4	5.8	6.5	---
EXPONENTIAL POWER	PWR	840	4 to 6	35.7	56.6	---
Floating Symbol Comparison	=F	329	3 to 5	0.13	0.26	---
	<>F	330				
	<F	331				
	<=F	332				
	>F	333				
	>=F	334				

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
FLOATING- POINT TO ASCII	FSTR	448	4 to 5	15.6	23.9	---
ASCII TO FLOATING-POINT	FVAL	449	3	21.2	31.4	---
MOVE FLOATING-POINT (SINGLE)	MOVF	469	3 to 4	0.18	0.20	---

A-2-14 Double-precision Floating-point Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DOUBLE SYMBOL COMPARISON	=D	335	3	5.1	6.7	---
	<>D	336				
	<D	337				
	<=D	338				
	>D	339				
	>=D	340				
DOUBLE FLOATING TO 16-BIT BINARY	FIXD	841	3	5.1	5.4	---
DOUBLE FLOATING TO 32-BIT BINARY	FIXLD	842	3	5.1	5.4	---
16-BIT BINARY TO DOUBLE FLOATING	DBL	843	3	3.5	4.3	---
32-BIT BINARY TO DOUBLE FLOATING	DBLL	844	3	3.5	4.3	---
DOUBLE FLOATING-POINT ADD	+D	845	4	6.0	7.1	---
DOUBLE FLOATING-POINT SUBTRACT	–D	846	4	6.1	7.1	---
DOUBLE FLOATING-POINT MULTIPLY	*D	847	4	6.1	7.1	---
DOUBLE FLOATING-POINT DIVIDE	/D	848	4	6.4	7.5	---
DOUBLE DEGREES TO RADIANS	RADD	849	3	6.1	6.5	---
DOUBLE RADIANS TO DEGREES	DEGD	850	3	6.0	6.4	---
DOUBLE SINE	SIND	851	3	14.7	21.5	0° specified
				20.4	35.4	45° specified
				18.5	35.0	90° specified
DOUBLE COSINE	COSD	852	3	14.1	20.6	0° specified
				19.6	29.9	45° specified
				19.1	29.8	90° specified
DOUBLE TANGENT	TAND	853	3	7.3	9.4	0° specified
				27.4	50.3	45° specified

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DOUBLE ARC SINE	ASIND	854	3	7.5	9.8	0° specified
				55.0	75.2	45° specified
				6.1	8.3	90° specified
DOUBLE ARC COSINE	ACOSD	855	3	8.3	10.9	0° specified
				55.9	72.8	45° specified
				43.7	72.8	90° specified
DOUBLE ARC TANGENT	ATAND	856	3	6.1	7.4	0° specified
				29.7	36.5	45° specified
DOUBLE SQUARE ROOT	SQRTD	857	3	16.6	23.4	---
DOUBLE EXPONENT	EXPD	858	3	39.7	58.4	---
DOUBLE LOGARITHM	LOGD	859	3	35.5	52.2	---
DOUBLE EXPONENTIAL POWER	PWRD	860	4	66	99	---

A-2-15 Table Data Processing Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SET STACK	SSET	630	3	7.6	9.4	Designating 5 words in stack area
				107	65	Designating 1,000 words in stack area
PUSH ONTO STACK	PUSH	632	3	4.9	5.9	---
FIRST IN FIRST OUT	FIFO	633	3	4.8	5.0	Designating 5 words in stack area
				231	167	Designating 1,000 words in stack area
LAST IN FIRST OUT	LIFO	634	3	5.3	7.1	---
DIMENSION RECORD TABLE	DIM	631	5	11.1	19.7	---
SET RECORD LOCATION	SETR	635	4	3.8	5.5	---
GET RECORD NUMBER	GETR	636	4	4.6	7.9	---
DATA SEARCH	SRCH	181	4	13.9	25.0	Searching for 1 word
				1940	3257	Searching for 1,000 words*1
SWAP BYTES	SWAP	637	3	10.1	17.5	Swapping 1 word
				1421	2098	Swapping 1,000 words*1
FIND MAXIMUM	MAX	182	4 to 5	4.8	5.8	Number of values being searched: 1
				465	672	Number of values being searched: 1,000*1

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DOUBLE FIND MAXIMUM	MAXL	174	4 to 5	4.8	5.8	Number of values being searched: 1
				465	773	Number of values being searched: 1,000*1
FIND MAXIMUM FLOATING	MAXF	176	4 to 5	5.2	6.5	Number of values being searched: 1
				682	1090	Number of values being searched: 1,000*1
FIND DOUBLE MAXIMUM FLOATING	MAXD	178	4 to 5	5.4	6.4	Number of values being searched: 1
				1435	2333	Number of values being searched: 1,000*1
FIND MINIMUM	MIN	183	4 to 5	4.8	5.8	Number of values being searched: 1
				465	677	Number of values being searched: 1,000*1
DOUBLE FIND MINIMUM	MINL	175	4 to 5	4.8	5.9	Number of values being searched: 1
				189	774	Number of values being searched: 1,000*1
FIND MINIMUM FLOATING	MINF	177	4 to 5	5.2	6.5	Number of values being searched: 1
				683	1091	Number of values being searched: 1,000*1
FIND DOUBLE MINIMUM FLOATING	MIND	179	4 to 5	5.2	6.4	Number of values being searched: 1
				1402	2303	Number of values being searched: 1,000*1
SUM	SUM	184	4	17.5	31.3	Adding 1 word
				900	1696	Adding 1,000 words*1
FRAME CHECKSUM	FCS	180	4	14.1	25.2	For 1-word table length
				1235	2089	For 1,000-word table length*1
STACK SIZE READ	SNUM	638	3	4.5	5.3	---
STACK DATA READ	SREAD	639	4	4.6	5.4	---
STACK DATA OVERWRITE	SWRIT	640	4	4.3	5.0	---
STACK DATA INSERT	SINS	641	4	8.2	9.3	---
				275	256	For 1,000-word table
STACK DATA DELETE	SDEL	642	4	6.1	7.8	---
				247	180	For 1,000-word table

*1 The instruction execution time is greatly affected by the amount to data. This will affect the cycle time. To reduce the effect on the cycle time, background execution can be specified. Refer to *10-2-5 Background Execution* for details.

A-2-16 Tracking Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
Unsigned One-word Record Search Instructions	RSRCH	360 to 364	6	13.9	15.9	Number of records: 1
				504	585	Number of records: 1,000
Unsigned Two-word Record Search Instructions	RSRCH2	370 to 374	6	14.7	17.6	Number of records: 1
				838	932	Number of records: 1,000
Unsigned Four-word Record Search Instructions	RSRCH4	380 to 384	6	17.0	19.1	Number of records: 1
				1544	1684	Number of records: 1,000
UNSIGNED ONE-WORD RECORD SORT	RSORT	203	5	149	156	100 records, split sorting disabled, sorting “99, 98, 97...0" to “0, 1, 2...99" (worst-case scenario)
UNSIGNED TWO-WORD RECORD SORT	RSORT2	204	5	250	249	
UNSIGNED FOUR-WORD RECORD SORT	RSORT4	205	5	457	440	

A-2-17 Data Control Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
PID CONTROL	PID	190	4	297	526	Initial execution
				234	423	Input ON and sampling
				71	117	Input ON and not sampling
				7.2	10.5	Input OFF
PID CONTROL WITH AUTOTUNING	PIDAT	191	4	302	600	Initial Execution
				237	428	Input ON and sampling
				73	118	Input ON and not sampling
				7.3	10.5	Input OFF
				120	203	Initial execution of autotuning
LIMIT CONTROL	LMT	680	4 to 5	10.8	18.3	---
DEAD BAND CONTROL	BAND	681	4 to 5	11.2	19.2	---
DEAD ZONE CONTROL	ZONE	682	4 to 5	10.9	17.7	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
TIME-PROPORTIONAL OUTPUT	TPO	685	4	6.9	10.2	Input OFF
				37	65	Input ON and duty specified or output limit disabled
SCALING	SCL	194	4	7.6	9.3	---
SCALING 2	SCL2	486	4	6.8	9.2	---
SCALING 3	SCL3	487	4	7.8	9.9	---
AVERAGE	AVG	195	4	22	40	Average of an operation
				212	351	Average of 64 operations

A-2-18 Subroutine Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SUBROUTINE CALL	SBS	091	2	0.90	2.8	---
SUBROUTINE ENTRY	SBN	092	2	2.8	4.1	---
SUBROUTINE RETURN	RET	093	1	0.43	2.0	---
MACRO	MCRO	099	4	16.8	21.7	---
GLOBAL SUBROUTINE RETURN	GSBS	750	2	0.90	2.8	---
GLOBAL SUBROUTINE CALL	GSBN	751	2	2.7	3.6	---
GLOBAL SUBROUTINE ENTRY	GRET	752	1	0.43	2.0	---

A-2-19 Interrupt Control Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SET INTERRUPT MASK	MSKS	690	3	10.6	22.1	---
READ INTERRUPT MASK	MSKR	692	3	9.6	14.8	---
CLEAR INTERRUPT	CLI	691	3	10.1	21.6	---
DISABLE INTERRUPTS	DI	693	1	10.3	20.4	---
ENABLE INTERRUPTS	EI	694	1	9.3	16.0	---

A-2-20 Step Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
STEP DEFINE	STEP	008	2	8.7	10.6	Step control bit ON
				8.7	9.8	Step control bit OFF
STEP START	SNXT	009	2	2.2	2.8	---

A-2-21 Basic I/O Unit Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
I/O REFRESH	IORF	097	3	10.1	12.2	1-word refresh (IN) for Basic I/O Units
				10.5	13.0	1-word refresh (OUT) for Basic I/O Units
SPECIAL I/O UNIT I/O REFRESH	FIORF	225	2	*1	*1	---
CPU BUS I/O REFRESH	DLNK	226	4	234	256	Allocated 1 word
7-SEGMENT DECODER	SDEC	078	4	2.5	3.3	---
DIGITAL SWITCH INPUT	DSW	210	6	24.8	39.6	4 digits, data input value: 0
				24.8	40.2	8 digits, data input value: 00
TEN KEY INPUT	TKY	211	4	7.2	9.7	Data input value: 00
				6.5	8.6	Data input value: FF
HEXADECIMAL KEY INPUT	HKY	212	5	25.9	40.8	Data input value: 00
				25.9	41.0	Data input value: FF
MATRIX INPUT	MTR	213	5	25.0	38.5	Data input value: 00
				25.0	38.5	Data input value: FF
7-SEGMENT DISPLAY OUTPUT	7SEG	214	5	31.4	51.9	4 digits
				34.6	59.4	8 digits
ANALOG INPUT DIRECT CONVERSION (for CJ1W-AD042)	AIDC	216	3	25.0	27.0	Analog input number: 1, Number of analog inputs used: 4
				38.8	41.6	Analog input number: 0, Number of analog inputs used: 4
ANALOG OUTPUT DIRECT CONVERSION (for CJ1W-DA042V)	AODC	217	3	23.1	24.4	Analog output number: 1, Number of analog outputs used: 4
				44.1	45.3	Analog output number: 0, Number of analog outputs used: 4

*1 Execution of the IORD, IORW, and FIORF instructions depends on the Special I/O Units for which they are being executed.

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
PCU HIGH-SPEED POSITIONING (CJ1W-NC□□4 or CJ1W-NC□81 only)	NCDMV	218	4	81.7	95.3	---
PCU POSITIONING TRIGGER (CJ1W-NC□81 only)	NCDTR	219	3	22.9	25.5	---
INTELLIGENT I/O READ	IORD	222	4	*1	*1	---
INTELLIGENT I/O WRITE	IOWR	223	4	*1	*1	---

*1 Execution of the IORD, IORW, and FIORF instructions depends on the Special I/O Units for which they are being executed.

A-2-22 Serial Communications Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
PROTOCOL MACRO	PMCR	260	5	57.8	97.8	Direct specification
				77	132	Operand specification, sending 1 word, receiving 1 word
PROTOCOL MACRO 2	PMCR2	264	6	49.5	96.0	Direct specification
				69	129	Operand specification, sending 1 word, receiving 1 word
TRANSMIT	TXD	236	4	57.5	93.8	Sending 1 byte
				517	947	Sending 256 bytes
RECEIVE	RXD	235	4	79	128	Storing 1 byte
				570	1033	Storing 256 bytes
TRANSMIT VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (CJ1W-SCU22, CJ1W-SCU32, or CJ1W-SCU42 only)	TXDU	256	4	75	130	Sending 1 byte
RECEIVE VIA SERIAL COMMUNICATIONS UNIT/SERIAL PORT (CJ1W-SCU22, CJ1W-SCU32, or CJ1W-SCU42 only)	RXDU	255	4	74	128	Storing 1 byte
DIRECT TRANSMIT VIA SERIAL COMMUNICATIONS UNIT	DTXDU	262	4	25.8	37.0	Sending 1 byte
				179	203	Sending 256 bytes
DIRECT RECEIVE VIA SERIAL COMMUNICATIONS UNIT	DRXDU	261	4	27.8	39.7	Storing 1 byte
				188	205	Storing 256 bytes
CHANGE SERIAL PORT SETUP	STUP	237	3	233	276	Addressed to COM port on CPU Unit

A-2-23 Network Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
NETWORK SEND	SEND	090	4	44.3	79.4	---
NETWORK SEND 2	SEND2	491	5	43.4	82.8	---
NETWORK RECEIVE	RECV	098	4	43.9	79.9	---
NETWORK RECEIVE 2	RECV2	492	5	44.5	82.8	---
DELIVER COMMAND	CMND	490	4	52.7	95.1	---
DELIVER COMMAND 2	CMND2	493	5	53.0	98.1	---
EXPLICIT MESSAGE SEND	EXPLT	720	4	78	134	---
EXPLICIT GET ATTRIBUTE	EGATR	721	4	74	127	---
EXPLICIT SET ATTRIBUTE	ESATR	722	3	69	117	---
EXPLICIT WORD READ	ECHRD	723	4	65	110	---
EXPLICIT WORD WRITE	ECHWR	724	4	64	110	---

A-2-24 File Memory Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
READ DATA FILE	FREAD	700	5	217	372	In binary
WRITE DATA FILE	FWRIT	701	5	216	366	In binary
WRITE TEXT FILE	TWRIT	704	5	205	370	---

A-2-25 Display Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
DISPLAY MESSAGE	MSG	046	3	6.9	10.5	Displaying message
				6.6	9.5	Deleting displayed message

A-2-26 Clock Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
CALENDAR ADD	CADD	730	4	15.6	22.5	---
CALENDAR SUB-TRACT	CSUB	731	4	16.4	24.9	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
HOURS TO SECONDS	SEC	065	3	3.6	4.1	---
SECONDS TO HOURS	HMS	066	3	3.5	4.0	---
CLOCK ADJUSTMENT	DATE	735	2	29.6	53.2	---

A-2-27 Debugging Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
TRACE MEMORY SAMPLING	TRSM	045	1	8.9	12.6	Sampling 1 bit and 0 words
				31.6	33.1	Sampling 31 bits and 6 words
				38.8	39.2	Sampling 31 bits and 16 words

A-2-28 Failure Diagnosis Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
FAILURE ALARM	FAL	006	3	7.9	14.7	Recording errors
				14.7	22.3	Deleting errors (in order of priority)
				12.9	22.5	Deleting errors (all errors)
				117	210	Deleting errors (individually)
SEVERE FAILURE ALARM	FALS	007	3	---	---	---
FAILURE POINT DETECTION	FPD	269	4	111	188	Bit address output, time monitored
				107	202	Bit address output, first error detection
				129	242	Message characters output, time monitored
				159	244	Message characters output, first error detection

A-2-29 Other Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SET CARRY	STC	040	1	0.048	0.060	---
CLEAR CARRY	CLC	041	1	0.048	0.060	---
SELECT EM BANK	EMBC	281	2	7.6	14.6	---
EXTEND MAXIMUM CYCLE TIME	WDT	094	2	7.6	17.1	---
SAVE CONDITION FLAGS	CCS	282	1	5.8	8.3	---
LOAD CONDITION FLAGS	CCL	283	1	6.4	9.9	---
CONVERT ADDRESS FROM CV	FRMCV	284	3	9.4	15.7	---
CONVERT ADDRESS TO CV	TOCV	285	3 to 4	10.3	18.2	---
DISABLE PERIPHERAL SERVICING	IOSP	287	1	---	---	---
ENABLE PERIPHERAL SERVICING	IORS	288	1	---	---	---

A-2-30 Block Programming Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
BLOCK PROGRAM BEGIN	BPRG	096	2	7.8	14.1	---
BLOCK PROGRAM END	BEND	801	1	8.8	13.4	---
BLOCK PROGRAM PAUSE	BPPS	811	2	5.4	8.4	---
BLOCK PROGRAM RESTART	BPRS	812	2	3.6	4.8	---
CONDITIONAL BLOCK EXIT	(Execution condition) EXIT	806	1	8.6	13.2	Block exited (input condition ON)
				2.0	2.6	Block not exited (input condition OFF)
CONDITIONAL BLOCK EXIT	EXIT (bit address)	806	2	9.8	14.8	Block exited (bit ON)
				3.6	4.2	Block not exited (bit OFF)
CONDITIONAL BLOCK EXIT (NOT)	EXIT NOT (bit address)	806	2	3.6	4.3	Block exited (bit OFF)
				8.9	14.9	Block not exited (bit ON)

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
Branching	IF (execution condition)	802	1	1.9	2.4	IF true (input condition ON)
				3.8	6.4	IF false (input condition OFF)
	IF (bit address)	802	2	3.2	4.0	IF true (bit ON)
				5.1	8.0	IF false (bit OFF)
Branching (NOT)	IF NOT (bit address)	802	2	5.1	8.2	IF true (bit OFF)
				3.2	4.1	IF false (bit ON)
Branching	ELSE	803	1	3.5	5.7	IF true
				5.3	7.3	IF false
Branching	IEND	804	1	5.3	8.5	IF true
				2.0	2.4	IF false
ONE CYCLE AND WAIT	WAIT (execution condition)	805	1	10.0	15.9	Do not wait (input condition ON)
				1.4	1.9	Wait (input condition OFF)
	WAIT (bit address)	805	2	9.2	13.5	Do not wait (bit ON)
				2.6	3.7	Wait (bit OFF)
ONE CYCLE AND WAIT (NOT)	WAIT NOT (bit address)	805	2	9.2	13.5	Do not wait (bit OFF)
				2.8	3.7	Wait (bit ON)
HUNDRED-MS TIMER WAIT	TIMW	813	3	15.6	22.9	Default setting
				16.0	23.2	Normal execution
	TIMWX	816	3	15.1	21.7	Default setting
				16.0	23.2	Normal execution
TEN-MS TIMER WAIT	TMHW	815	3	15.7	22.6	Default setting
				17.5	24.9	Normal execution
	TMHWX	817	3	15.2	22.1	Default setting
				16.4	23.4	Normal execution
COUNTER WAIT	CNTW	814	4	13.7	20.5	Default setting
				13.4	19.8	Normal execution
	CNTWX	818	4	13.1	19.5	Default setting
				13.5	19.7	Normal execution
Loop Control	LOOP	809	1	4.6	9.1	---
Loop Control	LEND (execution condition)	810	1	4.2	8.6	Do not loop (input condition ON)
				3.9	6.5	Loop (input condition OFF)
	LEND (bit address)	810	2	6.7	10.4	Do not loop (bit ON)
				6.6	8.2	Loop (bit OFF)
Loop Control (NOT)	LEND NOT (bit address)	810	2	6.7	10.9	Do not loop (bit OFF)
				6.6	8.2	Loop (bit ON)

A-2-31 Text String Processing Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
MOV STRING	MOV$	664	3	31.5	58.3	Transferring 1 character*1
CONCATENATE STRING	+$	656	4	56	104	1 character + 1 character*1
GET STRING LEFT	LEFT$	652	4	33.5	62.2	Retrieving 1 character from 2 characters*1
GET STRING RIGHT	RGHT$	653	4	33.4	62.1	Retrieving 1 character from 2 characters*1
GET STRING MIDDLE	MID$	654	5	32.3	60.8	Retrieving 1 character from 3 characters*1
FIND IN STRING	FIND$	660	4	30.3	56.3	Searching for 1 character from 2 characters*1
STRING LENGTH	LEN$	650	3	14.0	24.9	Detecting 1 character*1
REPLACE IN STRING	RPLC$	661	6	110	213	Replacing the first of 2 characters with 1 character*1
DELETE STRING	DEL$	658	5	45.6	86.8	Deleting the leading character of 2 characters*1
EXCHANGE STRING	XCHG$	665	3	40.3	75.4	Exchanging 1 character with 1 character*1
CLEAR STRING	CLR$	666	2	15.9	28.3	Clearing 1 character*1
INSERT INTO STRING	INS$	657	5	85	162	Inserting 1 character after the first of 2 characters*1
String Comparison Instructions	=$	670	4	27.0	50.9	Comparing 1 character with 1 character
	<>$	671				
	<$	672				
	<=$	673				
	>$	674				
	=$	675				

*1 The instruction execution time is greatly affected by the amount to data. This will affect the cycle time. To reduce the effect on the cycle time, background execution can be specified. Refer to *10-2-5 Background Execution* for details.

A-2-32 Task Control Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
TASK ON	TKON	820	2	12.5	19.0	Cyclic task specified
				13.6	22.7	Extra task specified
TASK OFF	TKOF	821	2	240	393	Cyclic task specified
				15.5	25.8	Extra task specified

A-2-33 Model Conversion Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
BLOCK TRANSFER	XFERC	565	4	6.7	8.2	Transferring 1 word
				362	409	Transferring 1,000 words
SINGLE WORD DISTRIBUTE	DISTC	566	4	4.6	5.3	Data distribute
				6.0	7.3	Stack operation
DATA COLLECT	COLLC	567	4	5.3	6.5	Data distribute
				4.5	13.0	Stack operation
				5.8	6.1	Stack operation 1 word FIFO Read
				42	142	Stack operation 1,000 word FIFO Read
MOVE BIT	MOVBC	568	4	4.9	5.7	---
BIT COUNTER	BCNTC	621	4	5.5	6.4	Counting 1 word
				873	974	Counting 1,000 words

A-2-34 Special Function Block Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
GET VARIABLE ID	GETID	286	4	7.6	12.5	---

A-2-35 SFC Instructions

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
STEP ACTIVATE	SA	784	2	9.1	11.8	---
STEP DEACTIVATE	SE	785	2	9.1	11.9	---
READ SET TIMER	TSR	780	3	3.9	4.2	---
SET STEP TIMER	TSW	781	3	5.7	8.2	---
SFC ON	SFCON	789	2	14.0	20.4	---

Instruction	Mnemonic	FUN No.	Length (steps)	ON execution time (μs)		Conditions
				CJ2H CPU6□(-EIP)	CJ2M-CPU□□	
SFC OFF	SFCOFF	790	2	249	402	---
SFC PAUSE WITH RESET	SFCPR	793	2	249	405	---
SFC PAUSE NO WITH RESET	SFCPRN	791	2	249	405	---

A-2-36 Function Block Instance Execution Time

Use the following equation to calculate the effect of instance execution on the cycle time when function block definitions have been created and the instances copied into the user program.

Effect of Instance Execution on Cycle Time
= Startup time (A)
+ I/O parameter transfer processing time (B)
+ Execution time of instructions in function block definition (C)

The following table shows the length of time for A, B, and C.

Operation			CPU Unit model	
			CJ2H-CPU6□(-EIP)	CJ2M-CPU□□
A	Startup time	Startup time not including I/O parameter transfer	3.3 μs	7.4 μs
B	I/O parameter transfer processing time The data type is indicated in parentheses.	1-bit I/O variable (BOOL)	0.24 μs	0.88 μs
		1-word I/O variable (INT, UINT, WORD)	0.19 μs	0.88 μs
		2-word I/O variable (DINT, UDINT, DWORD, REAL)	0.19 μs	1.2 μs
		4-word I/O variable (LINT, ULINT, LWORD, LREAL)	0.38 μs	2.96 μs
C	Function block definition instruction execution time	Total instruction processing time (same as standard user program)		

Example: CJ2H-CPU67-EIP
Input variables with a 1-word data type (INT): 3
Output variables with a 1-word data type (INT): 2
Total instruction processing time in function block definition section: 10 μs
Execution time for 1 instance = 3.3 μs + (3 + 2) × 0.19 μs + 10 μs = 14.25 μs

Note The execution time is increased according to the number of multiple instances when the same function block definition has been copied to multiple locations.



Additional Information

Number of Function Block Program Steps

Use the following equation to calculate the number of program steps when function block definitions have been created and the instances copied into the user program.

Number of steps
= Number of instances × (Call part size m + I/O parameter transfer part size n × Number of parameters) + Number of instruction steps in the function block definition p (See note.)

Note The number of instruction steps in the function block definition (p) will not be diminished in subsequence instances when the same function block definition is copied to multiple locations (i.e., for multiple instances). Therefore, in the above equation, the number of instances is not multiplied by the number of instruction steps in the function block definition (p).

Contents			CJ2 CPU Units
m	Call part		57 steps
n	I/O parameter transfer part The data type is shown in parentheses.	1-bit I/O variable (BOOL)	6 steps
		1-word I/O variable (INT, UINT, WORD)	6 steps
		2-word I/O variable (DINT, UDINT, DWORD, REAL)	6 steps
		4-word I/O variable (LINT, ULINT, LWORD, LREAL)	12 steps
p	Number of instruction steps in function block definition	The total number of instruction steps (same as standard user program) + 27 steps.	

Example:
Input variables with a 1-word data type (INT): 5
Output variables with a 1-word data type (INT): 5
Function block definition section: 100 steps
Number of steps for 1 instance = 57 + (5 + 5) × 6 steps + 100 steps + 27 steps = 244 steps

A-3 Auxiliary Area

A000 to A447: Read-only Area, A448 to A1000: Read/Write Area

A-3-1 Read-only Area (Set by System)

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A0	---	10-ms Incrementing Free Running Timer	This word contains the system timer used after the power is turned ON. A0 is set to 0000 hex when the power is turned ON and this value is automatically incremented by 1 every 10 ms. The value returns to 0000 hex after reaching FFFF hex (655,350 ms), and then continues to be automatically incremented by 1 every 10 ms. The timer will continue to be incremented when the operating mode is switched to RUN mode. Example: The interval can be counted between processing A and processing B without requiring timer instructions. This is achieved by calculating the difference between the value in A000 for processing A and the value in A000 for processing B. The interval is counted in 10 ms units.	Retained	Cleared	Every 10 ms after power is turned ON
A1	---	100-ms Incrementing Free Running Timer	This word contains the system timer used after the power is turned ON. A1 is set to 0000 hex when the power is turned ON and this value is automatically incremented by 1 every 100 ms. The value returns to 0000 hex after reaching FFFF hex (6,553,500 ms), and then continues to be automatically incremented by 1 every 100 ms. The timer will continue to be incremented when the operating mode is switched to RUN mode.	Retained	Cleared	Every 100 ms after power is turned ON
A2	---	1-s Incrementing Free Running Timer	This word contains a system timer used after the power is turned ON. A2 is set to 0000 hex when the power is turned ON and this value is automatically incremented by 1 every 1 s. The value returns to 0000 hex after reaching FFFF hex (65,535 s), and then continues to be automatically incremented by 1 every 1 s. The timer will continue to be incremented when the operating mode is switched to RUN mode.	Retained	Cleared	Every 1 s after power is turned ON
A50	A50.00 to A50.07	Basic I/O Unit Information, Rack 0 Slot 0	A bit will turn ON to indicate when the load short-circuit protection function alarm output has been given. Only the 4 most LSB are used for the CJ1W-OD202 (2 points per bit), only the LSB is used for the CJ1W-OD212, OD204, MD232 and only the two most LSB are used for the CJ1W-OD232. ON: Short circuited OFF: Normal	---	---	Refreshed each cycle.
	A50.08 to A50.15	Basic I/O Unit Information, Rack 0 Slot 1		---	---	
A51 to A69	A51.00 to A69.15	Basic I/O Unit Information, Racks 0 Slot 2 to Rack 3 Slot 9		---	---	
A90 to A93	---	User Program Date	These words contain in BCD the date and time that the user program was last overwritten. A90.00 to A90.07: Seconds (00 to 59) A90.08 to A90.15: Minutes (00 to 59) A91.00 to A91.07: Hour (00 to 23) A91.08 to A91.15: Day of month (01 to 31) A92.00 to A92.07: Month (01 to 12) A92.08 to A92.15: Year (00 to 99) A93.08 to A93.07: Day of the week (00: Sunday, 01: Monday, 02: Tuesday, 03: Wednesday, 04: Thursday, 05: Friday, 06: Saturday)	Retained	Retained	---
A94 to A97	---	Parameter Date	These words contain in BCD the date and time that the parameters were last overwritten. The format is the same as above	Retained	Retained	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A99	00	UM Read Protection Status	Indicates whether the entire user program in the PLC is read-protected. OFF: UM not read-protected. ON: UM read-protected.	Retained	Retained	When protection is set or cleared
	01	Task Read Protection Status	Indicates whether read protection is set for individual tasks. OFF: Tasks not read-protected. ON: Tasks read-protected.			---
	02	Program Write Protection Status when Read Protection Is Set	Indicates whether the program is write-protected. OFF: Write-enabled. ON: Write-protected.			
	03	Enable/Disable Status for Backing Up the Program to a Memory Card	Indicates whether creating a backup program file (.OBJ) is enabled or disabled. OFF: Enabled. ON: Disabled.			
	14	IR/DR Operation between Tasks	Turn ON this bit to share index and data registers between all tasks. Turn OFF this bit to use separate index and data registers between in each task. OFF: Independent ON: Shared (default)			---
A100 to A199	---	Error Log Area	When an error has occurred, the error code, error contents, and error's time and date are stored in the Error Log Area. Information on the 20 most recent errors can be stored. Each error record occupies 5 words; the function of these 5 words is as follows: First Word: Error code (bits 0 to 15) Second Word: Error contents (bits 0 to 15) Error contents: Address of Auxiliary Area word with details or 0000. Third Word: Minutes (bits 8 to 15), Seconds (bits 0 to 7) Seconds: 00 to 59, BCD Minutes: 00 to 59, BCD Fourth Word: Day of month (bits 8 to 15), Hours (bits 0 to 7) Hours: 00 to 23, BCD Day of month: 01 to 31, BCD Fifty Word: Year (bits 8 to 15), Month (bits 0 to 7) Year: 00 to 99, BCD Month: 00 to 12, BCD Errors generated by FAL(006) and FALS(007) will also be stored in this Error Log. The Error Log Area can be reset from the CX-Programmer. If the Error Log Area is full (20 records) and another error occurs, the oldest record in A100 to A104 will be cleared, the other 19 records will be shifted down, and the new record will be stored in A195 to A199.	Retained	Retained	Refreshed when error occurs. A50014 A300 A400

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A200	A200.11	First Cycle Flag	ON for one cycle after PLC operation begins (after the mode is switched from PROGRAM to RUN or MONITOR, for example). ON for the first cycle	---	---	---
	A200.12	Step Flag	ON for one cycle when step execution is started with STEP(008). This flag can be used for initialization processing at the beginning of a step. ON for the first cycle after execution of STEP(008).	Cleared	---	---
	A200.14	Task Started Flag	When a task switches from STANDBY or DISABLED to READY status, this flag will be turned ON within the task for one cycle only. ON: ON for first cycle (including transitions from STANDBY and DISABLED status) OFF: Other The only difference between this flag and A200.15 is that this flag also turns ON when the task switches from STANDBY to READY status.	Cleared	Cleared	---
	A200.15	First Task Startup Flag	ON when a task is executed for the first time. This flag can be used to check whether the current task is being executed for the first time so that initialization processing can be performed if necessary. ON: First execution OFF: Not executable for the first time or not being executed.	Cleared	---	---
A201	A201.10	Online Editing Wait Flag	ON when an online editing process is waiting. ON: Waiting for online editing OFF: Not waiting for online editing If another online editing command is received while waiting, the other command will not be recorded and an error will occur.	Cleared	Cleared	A527
	A201.11	Online Editing Flag	ON when an online editing process is being executed. ON: Online editing in progress OFF: Online editing not in progress	Cleared	Cleared	A527
A202	A202.00 to A202.07	Communications Port Enabled Flags	ON when a network instruction (SEND, RECV, CMND, PMCR, TXDU, or RXDU) or background execution can be executed with the corresponding port number. Bits 00 to 07 correspond to communications ports 0 to 7. ON: Network instruction is not being executed OFF: Network instruction is being executed (port busy) When two or more network instructions are programmed with the same port number, use the corresponding flag as an execution condition to prevent the instructions from being executed simultaneously. (The flag for a given port is turned OFF while a network instruction with that port number is being executed.) Cleared when an instruction is executed.	Cleared	---	---
	A202.08	CJ2 Instructions Enabled Flag	ON when CJ2 instructions can be used. This flag is ON by default. This flag can be used only with the following instructions: SEND2, CMND2, PMCR2, and RECV2.	---	Updated according to internal status (cleared).	---
	A202.15	Network Communications Port Allocation Enabled Flag	ON when there is a communications port available for automatic allocation when executing communications instructions (SEND, RECV, CMND, PMCR, TXDU, or RXDU). ON: Communications port available OFF: Communications port not available Use this flag to confirm whether a communications port is available for automatic allocation before executing communications instructions when using 9 or more communications instructions simultaneously.	Cleared	---	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A203 to A210	---	Communications Port Completion Codes	These words contain the completion codes for the corresponding port numbers when network instructions (SEND, RECV, CMND, or PMCR) or background execution have been executed. (The corresponding word will be cleared to 0000 hex when background execution has been completed.) Words A203 to A210 correspond to communications ports 0 to 7. Non-zero: Error code 0000: Normal condition The following codes will be stored when an Explicit Message Instruction (EXPLT, EGATR, ESATR, ECHRD, or ECHWR) has been executed. If the Explicit Communications Error Flag turns OFF, 0000 hex is stored. If the Explicit Communications Error Flag is ON and the Network Communications Error Flag is ON, the FINS end code is stored. If the Explicit Communications Error Flag is ON and the Network Communications Error Flag is OFF, the explicit message end code is stored. During communications, 0000 hex will be stored and the suitable code will be stored when execution has been completed. The code will be cleared when operation is started. (The completion code for a given port is cleared to 0000 when a network instruction with that port number is executed.) Cleared when an instruction is executed.	Cleared	---	---
A211	---	Number of Ports Available for CJ2 Network Communications Instruction	When the number of CJ2 network communications instructions that can be used reaches 0, A202.08 will turn OFF. This word can be used only with the following instructions: SEND2, CMND2, PMCR2, and RECV2. The contents of this word can be used to check communications traffic.	---	Updated according to internal status (cleared).	---
A213	A213.00 to A213.07	Explicit Communications Error Flag	Turn ON when an error occurs in executing an Explicit Message Instruction (EXPLT, EGATR, ESATR, ECHRD, or ECHWR). Bits 00 to 07 correspond to communications ports 0 to 7. ON: Error end OFF: Normal end The corresponding bit will turn ON both when the explicit message cannot be sent and when an error response is returned for the explicit message. The status will be maintained until the next explicit message communication is executed. The bit will always turn OFF when the next Explicit Message Instruction is executed.	Cleared	---	A219.00 to A219.07 A203 to A210
A214	A214.00 to A214.07	First Cycle Flags after Network Communications Finished	Each flag will turn ON for just one cycle after communications have been completed. Bits 00 to 07 correspond to ports 0 to 7. Use the Used Communications Port Number stored in A218 to determine which flag to access. 1: First cycle after communications finish only 2: Other status These flags are not effective until the next cycle after the communications instruction is executed. Delay accessing them for at least one cycle. Use the port number specified in A218 (Used Communications Port Numbers) to access the correct bit for the port being used.	Retained	Cleared	---
A215	A215.00 to A215.07	First Cycle Flags after Network Communications Error	Each flag will turn ON for just one cycle after a communications error occurs. Bits 00 to 07 correspond to ports 0 to 7. Use the Used Communications Port Number stored in A218 to determine which flag to access. Determine the cause of the error according to the Communications Port Completion Codes stored in A203 to A210. ON: First cycle after communications error only OFF: Other status These flags are not effective until the next cycle after the communications instruction is executed. Delay accessing them for at least one cycle. Use the port number specified in A218 (Used Communications Port Numbers) to access the correct bit for the port being used.	Retained	Cleared	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A216 and A217	---	Network Communications Completion Code Storage Address	The completion code for a communications instruction is automatically stored at the address with the I/O memory address given in these words. Place this address into an index register and use indirect addressing through the index register to read the communications completion code.	Retained	Cleared	---
A218	---	Used Communications Port Numbers	Stores the communications port numbers used when a communications instruction is executed using automatic communication port allocations. 0000 to 0007 hex: Communications port 0 to 7	Retained	Cleared	---
A219	A219.00 to A219.07	Communications Port Error Flags	ON when an error occurred during execution of a network instruction (SEND, RECV, CMND, or PMCR). Bits 00 to 07 correspond to communications ports 0 to 7. ON: Error occurred OFF: Normal condition	Retained	---	---
A220 to A259	A220.00 to 259.15	Basic I/O Unit Input Response Times	These words contain the actual input response times. 0 to 17 hexadecimal When the Basic I/O Unit input response time setting is changed in the PLC Setup while the PLC is in PROGRAM mode, the setting in the PLC Setup will not match the actual value in the Basic I/O Unit unless the power is turned OFF and then ON again. In that case, the actual value can be monitored in these words.	Retained	See function column.	PLC Setup (Basic I/O Unit Input response time settings)
A260	---	I/O Allocation Status	Indicates the current status of I/O allocation, i.e., Automatic I/O Allocation or User-set I/O Allocations. 0000 hex: Automatic I/O allocations BBBB hex: User-set I/O allocations	Retained	Retained	---
A261	A261.00	I/O Table Creation Error Details	ON: Error in CPU Bus Unit Setup Turns OFF when I/O tables are generated normally. ON: Error in CPU Bus Unit Setup OFF: I/O tables generated normally	Retained	Cleared	When I/O tables are generated
	A261.02		ON: Overflow in maximum number of I/O points Turns OFF when I/O tables are generated normally. ON: Overflow in maximum number of I/O points OFF: I/O tables generated normally			A401.11 (Too many I/O points)
	A261.03		ON: The same unit number was used more than once. Turns OFF when I/O tables are generated normally. ON: The same unit number was used more than once. OFF: I/O tables generated normally			A401.13 (duplicated number)
	A261.04		ON: I/O bus error Turns OFF when I/O tables are generated normally. ON: I/O bus error OFF: I/O tables generated normally			A401.14 (I/O bus error)
	A261.06		ON: I/O table error because a SYSMAC BUS Slave cannot be detected Turns OFF when I/O tables are generated normally. ON: SYSMAC BUS Slave missing OFF: I/O tables generated normally			---
	A261.07		ON: Error in a Special I/O Unit Turns OFF when I/O tables are generated normally. ON: Error in a Special I/O Unit OFF: I/O tables generated normally			---
	A261.09		ON: I/O detection has not been completed. Turns OFF when I/O tables are generated normally. ON: I/O detection has not been completed. OFF: I/O tables generated normally			---
A262 and A263	---	Maximum Cycle Time (0.1-ms increments)	These words contain the maximum cycle time since the start of PLC operation. The cycle time is recorded in 8-digit hexadecimal with the leftmost 4 digits in A263 and the rightmost 4 digits in A262. 0 to FFFFFFFF: 0 to 429,496,729.5 ms (0.1-ms increments)	Cleared	Cleared	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A264 and A265	---	Present Cycle Time (0.1-ms increments)	These words contain the present cycle time in increments of 0.1 ms. The time is recorded each cycle in 8-digit hexadecimal with the leftmost 4 digits in A264 and the rightmost 4 digits in A265. 00000000 to FFFFFFFF (0.0 to 429,496,729.5 ms)	Cleared	Cleared	---
A266 and A267	---	Present Cycle Time (0.01-ms increments)	These words contain the present cycle time in increments of 0.01 ms. The time is recorded each cycle in 8-digit hexadecimal with the leftmost 4 digits in A266 and the rightmost 4 digits in A267. 00000000 to FFFFFFFF (0.0 to 42,949,672.95 ms)	Cleared	Cleared	---
A293	---	Version Error Information	A value is set here when the transferred user program contains a function that is not supported by the unit version of the CPU Unit. 0000 hex: No error. 0001 hex: Error	Cleared	Cleared	Written at the start of operation
A294	---	Task Number when Program Stopped	This word contains the task number of the task that was being executed when program execution was stopped because of a program error. Normal tasks: 0000 to 007F hex (task 0 to 127) Interrupt tasks: 8000 to 80FF hex (task 0 to 255) A298 and A299 contain the program address where program execution was stopped.	Cleared	Cleared	A298/A299

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A295	A295.08	Instruction Processing Error Flag	This flag and the Error Flag (ER) will be turned ON when an instruction processing error has occurred and the PLC Setup has been set to stop operation for an instruction error. CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: Error Flag ON OFF: Error Flag OFF	Cleared	Cleared	PLC Setup (Operation when instruction error has occurred) The task number where the error is stored in A294 and the program address is stored in A298 and A299.
	A295.09	Indirect DM/EM BCD Error Flag	This flag and the Access Error Flag (AER) will be turned ON when an indirect DM/EM BCD error has occurred and the PLC Setup has been set to stop operation an indirect DM/EM BCD error. (This error occurs when the content of an indirectly addressed DM or EM word is not BCD although BCD mode has been selected.) CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: Not BCD OFF: Normal			
	A295.10	Illegal Access Error Flag	This flag and the Access Error Flag (AER) will be turned ON when an illegal access error has occurred and the PLC Setup has been set to stop operation an illegal access error. (This error occurs when a region of memory is access illegally.) CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. The following operations are considered illegal access: Reading/writing the system area Reading/writing EM File Memory Writing to a write-protected area Indirect DM/EM BCD error (in BCD mode) ON: Illegal access occurred OFF: Normal condition			
	A295.11	No END Error Flag	ON when there is not an END(001) instruction in each program within a task. CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: No END OFF: Normal condition			The task number where the error is stored in A294 and the program address is stored in A298 and A299.
	A295.12	Task Error Flag	ON when a task error has occurred. The following conditions generate a task error. There is not even one regular task that is executable (started). There is not a program allocated to the task. ON: Error OFF: Normal			
	A295.13	Differentiation Overflow Error Flag	The allowed value for Differentiation Flags which correspond to differentiation instructions has been exceeded. CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: Error OFF: Normal			
	A295.14	Illegal Instruction Error Flag	ON when a program that cannot be executed has been stored. CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: Error OFF: Normal			
	A295.15	UM Overflow Error Flag	ON when the last address in UM (User Memory) has been exceeded. CPU Unit operation will stop and the ERR/ALM indicator will light when this flag goes ON. ON: Error OFF: Normal			
A298 and A299	---	Program Address Where Program Stopped	These words contain the 8-digit binary program address of the instruction where program execution was stopped due to a program error.	Cleared	Cleared	(A294 contains the task number of the task where program execution was stopped.)

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A300	---	Error Log Pointer	When an error occurs, the Error Log Pointer is incremented by 1 to indicate the location where the next error record will be recorded as an offset from the beginning of the Error Log Area (A100 to A199). 00 to 14 hexadecimal The Error Log Pointer can be cleared to 00 by turning A500.14 (the Error Log Reset Bit) from OFF to ON. When the Error Log Pointer has reached 14 hex (20 decimal), the next record is stored in A195 to A199 when the next error occurs.	Retained	Retained	Refreshed when error occurs. A500.14
A301	---	Current EM Bank---	This word contains the current EM bank number in 4-digit hexadecimal. The current bank number can be changed with the EMBC(281) instruction. 0000 to 0018 hexadecimal	Cleared	Cleared	---
A302	A302.00 to A302.15	CPU Bus Unit Initializing Flags	These flags are ON while the corresponding CPU Bus Unit is initializing after its CPU Bus Unit Restart Bit (A501.00 to A501.15) is turned from OFF to ON or the power is turned ON. Bits 00 to 15 correspond to unit numbers 0 to 15. Use these flags in the program to prevent the CPU Bus Unit's refresh data from being used while the Unit is initializing. IORF(097) and FIORF(225) (CJ2 and CJ1H-R CPU Units only) cannot be executed while an CPU Bus Unit is initializing. OFF: Not initializing ON: Initializing (Reset to 0 automatically after initialization.) These bits are turned OFF automatically when initialization is completed.	Retained	Cleared	Written during initialization A501.00
A315	A315.12	Tag Memory Error Flag (Non-fatal Error) (CJ2H-CPU6□-EIP only.)	ON when an error occurs in the tag memory where network symbols are stored.	Retained	Cleared	---
	A315.13	Option Board Error Flag	Turns ON when the Option Board is removed while the power is being supplied or an Option Board that is not supported is mounted. OFF: No error, ON: Error	Cleared	Cleared	• Written when power is turned ON. • Refreshed each cycle during overseeing process.
	A315.15	Backup Memory Error Flag	ON when writing to the backup data area, source memory area, or comment memory area in the internal flash memory fails. This bit will turn OFF when writing is completed successfully.	Retained	Cleared	---
A330 to A335	A330.00 to A335.15	Special I/O Unit Initializing Flags	These flags are ON while the corresponding Special I/O Unit is initializing after its Special I/O Unit Restart Bit (A502.00 to A507.15) is turned from OFF to ON or the power is turned ON. The bits in these words correspond to unit numbers 0 to 95 as follows: A330.00 to A330.15: Units 0 to 15 A331.00 to A331.15: Units 16 to 31 ---- A335.00 to A335.15: Units 80 to 95 Use these flags in the program to prevent the Special I/O Unit's refresh data from being used while the Unit is initializing. Also, IORF(097) and FIORF(225) cannot be executed while a Special I/O Unit is initializing. OFF: Not initializing ON: Initializing (Reset to 0 automatically after initialization.) These bits are turned OFF automatically when initialization is completed.	Retained	Cleared	A502.00 to A507.15

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A336	A336.00 to A336.15	Units Detected at Startup (Racks 0 to 3)	The number of Units detected on each Rack is stored in 1-digit hexadecimal (0 to A hex). Rack 0: A336.00 to A336.03 Rack 1: A336.04 to A336.07 Rack 2: A336.08 to A336.11 Rack 3: A336.12 to A336.15 Example: The following would be stored if Rack 0 had 1 Unit, Rack 1 had 4 Units, Rack 2 had 8 Units and Rack 3 had 10 Units: A336 = A 8 4 1	Retained	Cleared	---
A339 and A340	---	Maximum Differentiation Flag Number	These words contain the maximum value of the differentiation flag numbers being used by differentiation instructions.	See Function column.	Cleared	Written at the start of operation A295.13
A343	A343.00 to A343.02	Memory Card Type	Indicates the type of Memory Card, if any, installed. 0: None 4: Flash ROM This information is recorded when the PLC power is turned ON or the Memory Card power switch is turned ON.	Retained	See Function column.	See Function column.
	A343.06	EM File Memory Format Error Flag	ON when a format error occurs in the first EM bank allocated for file memory. ON: Format error OFF: No format error The flag is turned OFF when formatting is completed normally.	Retained	Cleared	---
	A343.07	Memory Card Format Error Flag	ON when the Memory Card is not formatted or a formatting error has occurred. (The flag is turned OFF when formatting is completed normally.) ON: Format error OFF: No format error This flag is written when the PLC power is turned ON or the Memory Card power switch is turned ON.	Retained	See Function column.	See Function column.
	A343.08	File Transfer Error Flag	ON when an error occurred while writing data to file memory. ON: Error OFF: No error	Retained	Cleared	Refreshed when file data is written.
	A343.09	File Write Error Flag	ON when data cannot be written to file memory because it is write-protected or the data exceeds the capacity of the file memory. ON: Write not possible OFF: Normal condition	Retained	Cleared	Refreshed when file data is written.
	A343.10	File Read Error	ON when a file could not be read because of a malfunction (file is damaged or data is corrupted). ON: Read not possible OFF: Normal condition or read processing is being executed	Retained	Cleared	Refreshed when file data is read.
	A343.11	File Missing Flag	ON when an attempt is made to read a file that does not exist, or an attempt is made to write to a file in a directory that does not exist. ON: Specified file or directory is missing OFF: Normal condition or read processing is being executed	Retained	Cleared	Refreshed when file data is read.
	A343.13	File Memory Operation Flag	ON while any of the following operations is being executed. OFF when none of them are being executed. CMND instruction sending a FINS command to the local CPU Unit. Execution of a File Memory instruction. Program replacement using the control bit in the Auxiliary Area. Easy backup operation. ON: Instruction being executed. OFF: Instruction not being executed.	Retained	Cleared	Refreshed when file memory instruction is executed.
	A343.14	Accessing File Data Flag	ON while file data is being accessed. ON: File being accessed OFF: File not being accessed Use this flag to prevent two file memory instructions from being executed at the same time.	Retained	Cleared	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A343	A343.15	Memory Card Detected Flag	ON when a Memory Card has been detected. OFF when a Memory Card has not been detected. ON: Memory Card detected OFF: Memory Card not detected	Retained	Cleared	Refreshed when Memory Card is inserted, or the power is turned ON.
A344	A344.00 to A344.07	EM File Memory Starting Bank	Contains the starting bank number of EM trace memory or EM file memory. All EM banks from this starting bank to the last bank in the EM Area are formatted for use as file memory or trace memory. If EM Area banks are not formatted to file memory or trace memory, this A344 will be FFFF hex. To convert part of the EM Area to file memory or trace memory, select ***PLC - Memory Allocation - EM Memory Settings*** from the CX-Programmer, and then select File Memory or Trace Memory.	Retained	Retained	PLC Setup (EM File Setting Enabled parameter and EM Start File No. parameter)
	A344.14	EM Trace Memory Flag	When A344 is not FFFF hex and this flag in ON, the banks of the EM Area from the bank given in A344.00 to A344.07 to the end of the EM Area are formatted to trace memory.	Retained	Retained	
	A344.15	EM File Memory Flag	When A344 is not FFFF hex and this flag in ON, the banks of the EM Area from the bank given in A344.00 to A344.07 to the end of the EM Area are formatted to file memory.	Retained	Retained	
A345	A345.00	FB Program Source Information Flag	Turns ON when there is FB program source information in the source/comment memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	When transferred from the CX-Programmer
	A345.01	Symbol Table Information Flag	Turns ON when there is symbol table information in the source/comment memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	
	A345.02	Comment Information Flag	Turns ON when there is comment information in the source/comment memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	
	A345.03	Program Index Information Flag	Turns ON when there is program index information in the source/comment memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	
	A345.05	SFC Program Source Information Flag	Turns ON when there is SFC program source information in the source/comment memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	
	A345.07	Network Symbols (Tags) Information Flag	Turns ON when there is network symbols (tags) information in the tag memory. ON: Information present, OFF: Information not present	Retained	Internal status updated	
A346 and A347	---	Number of Remaining Words to Transfer	These words contain the 8-digit hexadecimal number of words remaining to be transferred by FREAD(700) or FWRIT(701). When one of these instructions is executed, the number of words to be transferred is written to A346 and A347. While the data is being transferred, the value in these words is decremented. A346 contains the rightmost 4-digits and A347 contains the leftmost 4-digits. Check the content of these words to determine whether or not the planned number of words have been transferred successfully.	Retained	Cleared	Written as FREAD or FWRIT is being executed. Decremented as data is actually transferred.
A351 to A354	---	Calendar/Clock Area	These words contain the CPU Unit's internal clock data in BCD. The clock can be set from the CX-Programmer, with the DATE(735) instruction, or with a FINS command (CLOCK WRITE, 0702). A351.00 to A351.07: Seconds (00 to 59) (BCD) A351.08 to A351.15: Minutes (00 to 59) (BCD) A352.00 to A352.07: Hours (00 to 23) (BCD) A352.08 to A352.15: Day of the month (01 to 31) (BCD) A353.00 to A353.07: Month (01 to 12) (BCD) A353.08 to A353.15: Year (00 to 99) (BCD) A354.00 to A354.07: Day of the week (00 to 06) (BCD) 00: Sunday, 01: Monday, 02: Tuesday, 03: Wednesday, 04: Thursday, 05: Friday, 06: Saturday	Retained	Retained	Written every cycle

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A360 to A391	A360.01 to A391.15	Executed FAL Number Flags	The flag corresponding to the specified FAL number will be turned ON when FAL(006) is executed. Bits A360.01 to A391.15 correspond to FAL numbers 001 to 511. ON: That FAL was executed OFF: That FAL wasn't executed The flag will be turned OFF when the error is cleared.	Retained	Cleared	Refreshed when error occurs. A402.15
A392	A392.04	Serial Port Error Flag	ON when an error has occurred at the RS-232C port. (Do not access this bit in Peripheral Bus Mode, 1:N NT Link mode, or Serial PLC Link Polling/Polled Unit mode.) ON: Error OFF: No error	Retained	Cleared	Refreshed when error occurs.
	A392.05	Serial Port Send Ready Flag (No-protocol mode)	ON when the RS-232C port is able to send data in no-protocol mode. ON: Able-to-send OFF: Unable-to-send	Retained	Cleared	Written after transmission
	A392.06	Serial Port Reception Completed Flag (No-protocol mode)	ON when the RS-232C port has completed the reception in no-protocol mode. • When the number of bytes was specified: ON when the specified number of bytes is received. • When the end code was specified: ON when the end code is received or 256 bytes are received.	Retained	Cleared	Written after reception
	A392.07	Serial Port Reception Overflow Flag (No-protocol mode)	ON when a data overflow occurred during reception through the RS-232C port in no-protocol mode. • When the number of bytes was specified: ON when more data is received after the reception was completed but before RXD(235) was executed. • When the end code was specified: ON when more data is received after the end code was received but before RXD(235) was executed. ON when 257 bytes are received before the end code. ON: Overflow OFF: No overflow	Retained	Cleared	
A393	A393.00 to A393.07	Serial Port PT Communications Flag	The corresponding bit will be ON when the serial port is communicating in NT Link Mode or in Serial PLC Link Mode. Bits 0 to 7 correspond to units 0 to 7. ON: Communicating OFF: Not communicating	Retained	Cleared	Refreshed when there is a normal response to the token.
	A393.08 to A393.15	Serial Port PT Priority Registered Flags	The corresponding bit will be ON for the PT that has priority when the RS-232C port is communicating in NT link mode. Bits 0 to 7 correspond to units 0 to 7. These flags are written when the priority registration command is received. ON: Priority registered OFF: Priority not registered	Retained	Cleared	See Function column.
	A393.00 to A393.15	Serial Port Reception Counter (No-protocol mode)	Indicates (in binary) the number of bytes of data received when the RS-232C port is in no-protocol mode.	Retained	Cleared	Refreshed when data is received.

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A395	A395.06	File Deleted Flags	The system deleted the remainder of a Memory Card file that was being updated when a power interruption occurred. ON: File deleted OFF: No files deleted	Cleared	Cleared	Refreshed when the system deletes the file.
	A395.07		The system deleted the remainder of an EM file memory file that was being updated when a power interruption occurred. ON: File deleted OFF: No files deleted	Cleared	Cleared	Refreshed when the system deletes the file.
	A395.10	ER/AER Flag for Background Execution	ON when an instruction processing error or an illegal area access error occurs during background processing. ON: Error. OFF (0) when power is turned ON. OFF (0) when operation starts. OFF: No errors. OFF (0) when background processing starts.	Cleared	Cleared	---
	A395.11	Memory Corruption Detected Flag	ON when memory corruption is detected when the power supply is turned ON. ON: Memory corruption OFF: Normal operation	Retained	See Function column.	Refreshed when power is turned ON.
	A395.12	DIP Switch Pin 6 Status Flag	The status of pin 6 on the DIP switch on the front of the CPU Unit is written to this flag every cycle. ON: Pin 6 ON OFF: Pin 6 OFF	Retained	See Function column.	Written every cycle.
A400	---	Error code	When a non-fatal error (user-defined FALS(006) or system error) or a fatal error (user-defined FALS(007) or system error) occurs, the 4-digit hexadecimal error code is written to this word. (Refer to *A-3-3 Details on Auxiliary Area Operation*) on page A-138. When two or more errors occur simultaneously, the highest error code will be recorded.	Cleared	Cleared	Refreshed when error occurs.

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A401	A401.03	Card Transfer Error Flag	• ON when the contents of the Memory Card were not correctly read into the CPU Unit when the power was turned ON. • ON when a program file (.OBJ) that includes network symbols is transferred when the power is turned ON (CJ2H-CPU6□ and CJ2M-CPU1□ only.)	Cleared for error clear operation.	Cleared	---
	A401.05	Version Error Flag	ON when the unit version of the CPU Unit might not support the user program that was transferred.	Cleared for error clear operation.	Cleared	---
	A401.06	FALS Error Flag (Fatal error)	ON when a non-fatal error is generated by the FALS(006) instruction. The CPU Unit will continue operating and the ERR/ALM indicator will flash. The corresponding error code will be written to A400. Error codes C101 to C2FF correspond to FALS numbers 001 to 511. ON: FALS(006) executed OFF: FALS(006) not executed This flag will be turned OFF when the FALS errors are cleared.	Cleared	Cleared	Refreshed when error occurs. A400
	A401.08	Cycle Time Exceeded Flag (Fatal error)	ON if the cycle time exceeds the maximum cycle time set in the PLC Setup (the cycle time monitoring time). CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. OFF: Cycle time under max. ON: Cycle time over max. This flag will be turned OFF when the error is cleared.	Cleared	Cleared	Refreshed when the cycle time exceeds maximum. PLC Setup (Cycle time monitoring time)
	A401.09	Program Error Flag (Fatal error)	ON when program contents are incorrect. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. The task number where the error occurred will be stored in A294 and the program address will be stored in A298 and A299. The type of program error that occurred will be stored in bits 8 to 15 of A295. Refer to the description of A295 and to the *Programming Manual* for more details on program errors. ON: Error OFF: No error This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A294, A295, A298 and A299
	A401.10	I/O Setting Error Flag (Fatal error)	• ON when the registered I/O tables do not match actual I/O tables (i.e., when the registered Units do not match the Units that are actually connected). • With the CJ2H-CPU6□-EIP, ON when an Interrupt Input Unit is mounted in a slot other than those shown below. CJ2H-CPU6□-EIP: CPU Rack slots 0 to 3 CJ2H-CPU6□ or CJ2M-CPU□□: CPU Rack slots 0 to 4 • ON when the registered I/O tables for a CJ2H-CPU6□-EIP or CJ2M-CPU1□ CPU Unit are downloaded to a CJ2H-CPU6□ or CJ2M-CPU1□ CPU Unit, or when the registered I/O tables for a CJ2H-CPU6□ CPU Unit are downloaded to a CJ2H-CPU6□-EIP or CJ2M-CPU3□ CPU Unit. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. ON: Error OFF: No error This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A405.08
	A401.11	Too Many I/O Points Flag (Fatal error)	ON when the number of I/O points being used in Basic I/O Units exceeds the maximum allowed for the PLC or when there are more than 11 Units connected in one Rack. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. ON: Error OFF: No error This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A407

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A401	A401.13	Duplication Error Flag (Fatal error)	ON in the following cases: • Two CPU Bus Units have been assigned the same unit number. • Two Special I/O Units have been assigned the same unit number. • Two Basic I/O Units have been allocated the same data area words. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. The duplicated unit number is indicated in A409 to A416. ON: Duplication error OFF: No duplication This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A410 to A416
	A401.14	I/O Bus Error Flag (Fatal error)	ON when an error occurs in a data transfer between the CPU Unit and a Unit mounted to a slot or when the End Cover is not connected to the CPU Rack or an Expansion Rack. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The slot number (00 to 09) where the I/O bus error occurred is written to A404.00 to A404.07 in binary. If an End Cover is not connected, 0E hex will be stored. The rack number (00 to 07) where the I/O bus error occurred is written to A404.08 to A404.15 in binary. These bits will contain 0B hex if an I/O bus error occurs on a CJ2HCPU6□-EIP or CJ2M-CPU3□ built-in network. If an End Cover is not connected, 0E hex will be stored.	Cleared	Cleared	A404
	A401.15	Memory Error Flag (Fatal error)	ON when an error occurred in memory or there was an error in automatic transfer from the Memory Card when the power was turned ON. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. ON: Error OFF: No error The location where the error occurred is indicated in A403.00 to A403.08, and A403.09 will be turned ON if there was an error during automatic transfer at startup. This flag will be turned OFF when the error is cleared. The automatic transfer at startup error cannot be cleared without turning OFF the PLC.	Cleared	Cleared	A403

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A402	A402.00	Other Non-fatal Error Flag (System Work Memory Error)	• ON when a non-fatal error other than a non-fatal error allocated to A402.01 to A402.15 occurs. (E.g., When an error occurs in memory for online editing.) • Details of the other non-fatal errors are stored in A315.	Cleared for error clear operation.	Cleared	---
	A402.02	Special I/O Unit Setting Error Flag (Non-fatal error)	ON when an installed Special I/O Unit does not match the Special I/O Unit registered in the I/O table. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The unit number of the Unit where the setting error occurred is indicated in A428 to A433. ON when any of the following occur for a Unit registered in the synchronous unit operation settings. • The Unit does not support the synchronous unit operation function. • The Unit is not connected in the PLC. • The Unit is not in the CPU Rack (i.e., it is in an Expansion Rack). The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error This flag will be turned OFF when the error is cleared and the correct PLC Setup is transferred to the CPU Unit. The unit number of the Unit where the setting error occurred is indicated in A428 to A433.	Cleared	Cleared	A428 to A433
	A402.03	CPU Bus Unit Setting Error Flag (Non-fatal error)	ON when an installed CPU Bus Unit does not match the CPU Bus Unit registered in the I/O table. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The unit number of the Unit where the setting error occurred is written to A427	Cleared	Cleared	A427

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A402	A402.04	Battery Error Flag (Non-fatal error)	ON if the CPU Unit's battery is disconnected or its voltage is low and the Detect Battery Error setting has been set in the PLC Setup. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. This flag can be used to control an external warning light or other indicator to indicate that the battery needs to be replaced.	Cleared	Cleared	PLC Setup (Detect Battery Error)
	A402.06	Special I/O Unit Error Flag (Non-fatal error)	ON when an error occurs in a data exchange between the CPU Unit and a Special I/O Unit (including an error in the Special I/O Unit itself). The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. The Special I/O Unit where the error occurred will stop operating. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The unit number of the Unit where the data exchange error occurred is indicated in A418 through A423.	Cleared	Cleared	A418 to A423
	A402.07	CPU Bus Unit Error Flag (Non-fatal error)	ON when an error occurs in a data exchange between the CPU Unit and an CPU Bus Unit (including an error in the CPU Bus Unit itself). The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. The CPU Bus Unit where the error occurred will stop operating. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The unit number of the Unit where the data exchange error occurred is indicated in A417.	Cleared	Cleared	A417
	A402.10	PLC Setup Error Flag (Non-fatal error)	ON when there is a setting error in the PLC Setup. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. The location of the error will be written to A406. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. It will also turn OFF for a setting error for synchronous unit operation settings if correct settings are transferred to the Synchronous Unit.	Cleared	Cleared	A406
	A402.12	Basic I/O Unit Error Flag (Non-fatal error)	ON when an error has occurred in a Basic I/O Unit. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error This flag will be turned OFF when the error is cleared. The location of the error will be written to A408.	Cleared	Cleared	A408
	A402.13	Duplicate Refresh Error Flag (Non-fatal error)	ON when the Detect Duplicate Refresh Errors parameter in the PLC Setup is set to detect duplicate refresh errors and one of the following occurs for the same Special I/O Unit. • FIORF(225), IORF(097), IORD(222) or IOWR(223) in a cyclic task is competing with FIORF(225), IORF(097), IORD(222) or IOWR(223) in an interrupt task. • FIORF(225), IORF(097), IORD(222) or IOWR(223) was executed in an interrupt task when I/O was being refreshed. If cyclic refreshing is not disabled in the PLC Setup for a Special I/O Unit and FIORF(225), IORF(097), IORD(222) or IOWR(223) is executed for the same Special I/O Unit in an interrupt task, a duplicate refresh error will occur.	Cleared	Cleared	A426, PLC Setup (Detect Duplicate Refresh Errors parameter in the PLC Setup is set to "Detect")

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A402	A402.15	FAL Error Flag (Non-fatal error)	ON when a non-fatal error is generated by executing FAL(006). The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. The bit in A360 to A391 that corresponds to the FAL number specified in FALS(006) will be turned ON and the corresponding error code will be written to A400. Error codes 4101 to 42FF correspond to FAL numbers 001 to 2FF (0 to 511). ON: FALS(006) error occurred OFF: FALS(006) not executed This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A360 to A391, A400
A403	A403.00 to A403.08	Memory Error Location	When a memory error occurs, the Memory Error Flag (A401.15) is turned ON and one of the following flags is turned ON to indicate the memory area where the error occurred. A403.00: User program A403.04: PLC Setup A403.05: Registered I/O Table A403.07: Routing Table A403.08: CPU Bus Unit Settings When a memory error occurs, the CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error The corresponding flag will be turned OFF when the error is cleared.	Cleared	Cleared	A401.15
A404	A404.00 to A404.07	I/O Bus Error Slot Number	Contains the 8-bit binary slot number (00 to 09) where an I/O Bus Error occurred. If an I/O bus error occurs in the CJ2H-CPU6□-EIP built-in EtherNet/IP section, 0B hex will be stored. When the End Cover is not connected to the CPU Rack or an Expansion Rack, F hex will be stored. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. 00 to 09 (slot number 00 to 09) The I/O Bus Error Flag (A401.14) will be ON. This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A401.14
	A404.08 to A40.415	I/O Bus Error Rack Number	Contains the 8-bit binary rack number (00 to 03) where an I/O Bus Error occurred. If an I/O bus error occurs in the CJ2H-CPU6□-EIP built-in EtherNet/IP section, 0B hex will be stored. When the End Cover is not connected to the CPU Rack or an Expansion Rack, F hex will be stored. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. 00 to 03 (rack number 00 to 03) The I/O Bus Error Flag (A401.14) will be ON. This flag will be turned OFF when the error is cleared.	Cleared	Cleared	A401.14
A405	A405.08	Interrupt Input Unit Position Error Flag	ON when the Interrupt Input Unit is not connected in the following range. If this flag turns ON, an I/O setting error will occur (fatal error), and operation will stop. • CJ2H-CPU6□-EIP: CPU Rack slots 0 to 3 (the 4 Unit on the right of the CPU Unit) • CJ2H-CPU6□: CPU Rack slots 0 to 4 (the 5 Units on the right of the CPU Unit) An error will also occur if the Unit is physically mounted in the range given above but it is not allocated in this range in the I/O tables with a dummy unit registration.	Cleared	Cleared	A401.10
	A405.15	Peripheral Servicing Too Long Flag	Turns ON when the peripheral servicing time in a Parallel Processing Mode exceeds 2 s. This will also cause a cycle time error and operation will stop. ON: Too long OFF: Not too long	Cleared	Cleared	A268

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A407	A407.00 to A407.12	Too Many I/O Points, Details	When any of the following values overflows or an duplication error occurs, the corresponding value will be stored as binary data. The operation of the CPU Unit will stop. The ERR/ALM/ indicator on the front of the CPU Unit will light. For information on the reason the error occurred, refer to Too Many I/O Points, Cause (A407.13 to A407.15). 1. The number of I/O points will be written here when the total number of I/O points set in the I/O Table (excluding Slave Racks) exceed the maximum allowed for the CPU Unit. 2. The number of interrupt input points when the number of interrupt input points exceeds 32. 3. The number of Racks will be written here when the number of Expansion Racks exceeds the maximum. The relevant value will be written here (A407.00 to A407.12) when the error occurs. These bits will be cleared when the error is cleared.	Cleared	Cleared	A401.11, A407.13 to A407.15
	A407.13 to A407.15	Too Many I/O Points, Cause	The 3-digit binary value of these bits indicates the cause of the Too Many I/O Points Error and shows the meaning of the value written to bits A407.00 to A407.12. Values of 000 to 101 (0 to 5) correspond to causes 1 through 6 described in “Too Many I/O Points, Cause 1,” above. 000: Too many I/O total 001: Too many interrupt input points 101: Too many Racks 111: Too many Units on a Rack These bits will be cleared when the error is cleared.	Cleared	Cleared	---
A408	A408.00 to A408.07	Basic I/O Unit Error, Slot Number	When an error has occurred in a Basic I/O Unit, A402.12 will be turned ON and the slot number where the error occurred will be written here in binary. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. 00 to 09 hexadecimal (Slots 0 to 9) These bits will be cleared when the error is cleared.	Cleared	Cleared	A402.12
	A408.08 to A408.15	Basic I/O Unit Error, Rack Number	When an error has occurred in a Basic I/O Unit, A402.12 will be turned ON and the Rack number where the error occurred will be written here in binary. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. 00 to 03 hexadecimal (Racks 0 to 3) These bits will be cleared when the error is cleared.	Cleared	Cleared	A402.12
A409	A409.00 to A409.03	Expansion Rack Number Duplication Flags	The corresponding flag will be turned ON when an Expansion Rack's starting word address was set from the CX-Programmer and two Racks have overlapping word allocations or a Rack's starting address exceeds CIO 0901. Bits 00 to 03 correspond to Racks 0 to 3. ON: Same words allocated to two different Racks or Rack starting address exceeds CIO 0901. OFF: No error The corresponding flag will be cleared when the error is cleared.	Cleared	Cleared	---
A410	A410.00 to A410.15	CPU Bus Unit Number Duplication Flags	The Duplication Error Flag (A401.13) and the corresponding flag in A410 will be turned ON when an CPU Bus Unit's unit number has been duplicated. Bits 00 to 15 correspond to unit numbers 0 to F. CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. ON: Duplication detected OFF: No duplication	Cleared	Cleared	A401.13

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A411 to A416	A411.00 to A416.15	Special I/O Unit Number Duplication Flags	The Duplication Error Flag (A401.13) and the corresponding flag in A411 through A416 will be turned ON when a Special I/O Unit's unit number has been duplicated. Bits 00 to 15 correspond to unit numbers 0 to F. (Bits A411.00 to A416.15 correspond to unit numbers 000 to 05F (0 to 95).) CPU Unit operation will stop and the ERR/ALM indicator on the front of the CPU Unit will light. The corresponding bit will also be turned ON when the Special I/O Unit's words are also allocated to a Basic I/O Unit on an Expansion Rack because of the Expansion Rack's starting word setting. ON: Duplication detected OFF: No duplication	Cleared	Cleared	A401.13
A417	A417.00 to A417.15	CPU Bus Unit Error, Unit Number Flags	When an error occurs in a data exchange between the CPU Unit and an CPU Bus Unit, the CPU Bus Unit Error Flag (A402.07) is turned ON and the bit in A417 corresponding to the unit number of the Unit where the error occurred is turned ON. Bits 00 to 15 correspond to unit numbers 0 to F. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error	Cleared	Cleared	A402.07
A418 to A423	A418.00 to A423.15	Special I/O Unit Error, Unit Number Flags	When an error occurs in a data exchange between the CPU Unit and a Special I/O Unit, the Special I/O Unit Error Flag (A402.06) will be turned ON. Each bit corresponds to a unit number. Bit 00 in A418 to bit 15 in A423 correspond to unit numbers 0 to 95. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Error OFF: No error The flag will be turned OFF when the error is cleared. If the unit number of the Unit is uncertain, none of the flags will be turned ON.	Cleared	Cleared	A402.06
A426	A426.00 to A426.11	Duplicate Refresh Error Unit Number	When A426.13 is ON, these bits: Contain the unit number of the Special I/O Unit for which duplicate refreshing was performed. These bits will be cleared when the error is cleared. Unit number: 000 to 05F (0 to 95) The flag will be turned OFF when the error is cleared.	Cleared	Cleared	A402.13 A426.15
	A426.15	Duplicate Refresh Error Cause	When A402.13 (the Duplicate Refresh Error Flag) is ON, this flag indicates the cause of the error. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Duplicated refreshing	Cleared	Cleared	A402.13, A426.00 to A426.11
A427	A427.00 to A427.15	CPU Bus Unit Setting Error, Unit Number Flags	When an CPU Bus Unit Setting Error occurs, A402.03 and the bit in this word corresponding to the Unit's unit number are turned ON. Bits 00 to 15 correspond to unit numbers 0 to F. The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash. ON: Setting error OFF: No setting error	Cleared	Cleared	Refreshed when power is turned ON or I/O is recognized. A402.03
A428 to A433	A428.00 to A433.15	Special I/O Unit Setting Error, Unit Number Flags	When a Special I/O Unit Setting Error occurs, A402.02 and the bit in these words corresponding to the Unit's unit number are turned ON. Bits 00 to 15 correspond to unit numbers 0 to F. ON: Setting error OFF: No setting error The CPU Unit will continue operating and the ERR/ALM indicator on the front of the CPU Unit will flash.	Cleared	Cleared	Refreshed when power is turned ON or I/O is recognized. A402.02

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A440	---	Max. Interrupt Task Processing Time	Contains the Maximum Interrupt Task Processing Time in units of 0.1 ms. (This value is written after the interrupt task with the max. processing time is executed and cleared when PLC operation begins.) Disabled when high-speed interrupt function is enabled in the PLC Setup.	Cleared	Cleared	See Function column.
A441	---	Interrupt Task With Max. Processing Time	Contains the task number of the interrupt task with the maximum processing time. Hexadecimal values 8000 to 80FF correspond to task numbers 00 to FF. Bit 15 is turned ON when an interrupt has occurred. (This value is written after the interrupt task with the max. processing time is executed and cleared when PLC operation begins.) Disabled when high-speed interrupt function is enabled in the PLC Setup.	Cleared	Cleared	See Function column.
A446	A446.00 to A446.07	Number of Times Protection Has Been Disabled	• Counts up each time protection disable fails (i.e., due to the protection disable password being input incorrectly). • Displays the total number of times that protection was disabled for UM protection and task protection. • The counter stops counting when it reaches 255 (decimal). • When all protection has been disabled, the counter will be set to 00 hex.	---	---	---
A450	---	CIO Area Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A451	---	Word Area Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A452	---	Holding Area Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A459	---	Index Register Area Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A460	---	DM Area Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A461	---	EM Bank 0 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A462	---	EM Bank 1 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A463	---	EM Bank 2 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A464	---	EM Bank 3 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A465	---	EM Bank 4 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A466	---	EM Bank 5 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A467	---	EM Bank 6 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A468	---	EM Bank 7 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A469	---	EM Bank 8 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A470	---	EM Bank 9 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A471	---	EM Bank A Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A472	---	EM Bank B Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A473	---	EM Bank C Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A474	---	EM Bank D Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A475	---	EM Bank E Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A476	---	EM Bank F Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A477	---	EM Bank 10 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---

Address		Name	Function	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits					
A478	---	EM Bank 11 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A479	---	EM Bank 12 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A480	---	EM Bank 13 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A481	---	EM Bank 14 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A482	---	EM Bank 15 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A483	---	EM Bank 16 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A484	---	EM Bank 17 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A485	---	EM Bank 18 Designation	This word stores a fixed code that is used to specify the area when an address is specified as a parameter for a function block instead of an input-output variable (VER_IN_OUT). This code is used only by the OMRON FB Library. Do not change the contents of this word.	Fixed value is set.	Fixed value is set.	---
A10001 to A10003	---	Manufacturing Lot Number	The manufacturing lot number of the CPU Unit is stored as BCD data. Hardware can be identified by referring to this area. hex A10000.00 to A10000.07: 00 hex A10000.08 to A10000.15: 00 hex A10001.00 to A10000.07: 00 hex A10001.08 to A10001.15: Day (01 to 31) A10002.00 to A10002.07: Month (01 to 12) A10002.08 to A10002.15: Year (00 to 09) A10003.00 to A10003.07: 00 hex A10003.08 to A10003.15: 00 hex	---	---	---
A10100	A10100.00	Synchronous Unit Operation Servicing Flag	ON while synchronous unit operation is being performed. ON: Synchronous unit operation being performed OFF: Synchronous unit operation not being performed	Cleared	Cleared	Refreshed when the synchronous signal turns ON the second time after power is turned ON or Unit is restarted.
A10101	---	Synchronous Input Data Refresh Error Code	This word contains 0001 hex when the CPU Unit fails to receive synchronous input data from a Synchronous Unit within the specified time. 0001 hex: Error 0000 hex: Normal	Cleared	Cleared	When synchronous data is refreshed.
A10102	---	Synchronous Operation Cycle Time	This word contains the synchronous operation cycle time set in the PLC Setup.	Retained	Retained	When power is turned ON or the Unit is restarted.

Note In CJ-series PLCs, the following flags are provided in a special read-only area and can be specified with the labels given in the table. These flags are not contained in the Auxiliary Area. Refer to *6-20 Condition Flags* and *6-21 Clock Pulses* for details.

Error Flag	Condition Flag Area
Access Error Flag	
Carry Flag	
Greater Than Flag	
Equals Flag	
Less Than Flag	
Negative Flag	
Overflow Flag	
Underflow Flag	
Greater Than or Equals Flag	
Not Equal Flag	
Less than or Equals Flag	
Always ON Flag	
Always OFF Flag	
0.02-s clock pulse	Clock Pulse Area
0.1-s clock pulse	
0.2-s clock pulse	
1-s clock pulse	
1-min clock pulse	

A-3-2 Read/Write Area (Set by User)

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A500	A500.12	IOM Hold Bit	Turn this bit ON to preserve the status of the I/O Memory when shifting from PROGRAM to RUN or MONITOR mode or vice versa. (If the status of the IOM Hold Bit itself is preserved in the PLC Setup (IOM Hold Bit Status), the status of the I/O Memory Area will be retained when the PLC is turned ON or power is interrupted.)	ON: Retained OFF: Not retained	Retained	Cleared (PLC Setup can be set to retain status.)	See Function column. PLC Setup (IOM Hold Bit Status setting)
	A500.13	Forced Status Hold Bit	Turn this bit ON to preserve the status of bits that have been force-set or force-reset when shifting from PROGRAM to MONITOR mode or vice versa, or when turning ON the power. Be sure to always use this bit together with the IOM Hold Bit (A500.12). Turn ON the IOM Hold Bit (A500.12) at the same time as this bit. Status will also be held with the power supply is interrupted. To preserve the status of this bit when the PLC is turned ON, the Forced Status Hold Bit Startup Hold Setting parameter in the PLC Setup must be turned set.	ON: Retained OFF: Not retained	Retained	Cleared (PLC Setup can be set to retain status.)	See Function column. PLC Setup (Forced Status Hold Bit Status setting)
	A500.14	Error Log Reset Bit	Turn this bit ON to reset the Error Log Pointer (A300) to 00. The contents of the Error Log Area itself (A100 to A199) are not cleared. This bit is automatically reset to 0 after the Error Log Pointer is reset.	OFF → ON: Clear	Retained	Cleared	A100 to A199, A300
	A500.15	Output OFF Bit	Turn this bit ON to turn OFF all outputs from Basic I/O Units, regardless of the status of the output bits in I/O memory (the same as when a fatal error occurs). The INH indicator on the front of the CPU Unit will light. **Note:** The status of this bit is held when the power supply is turned OFF. The status of outputs bits allocated to Special I/O Units in the CIO Area (e.g., external analog outputs) depends on the functions of the individual Special I/O Units.	---	Retained	Retained	---
A501	A501.00 to A501.15	CPU Bus Unit Restart Bits	Turn these bits ON to restart (initialize) the CPU Bus Unit with the corresponding unit number. Bits 00 to 15 correspond to unit numbers 0 to F. When a restart bit is turned ON, the corresponding CPU Bus Unit Initializing Flag (A302.00 to A302.15) will be turned ON. Both the restart bit and initializing flag will be turned OFF automatically when initialization is completed.	OFF to ON: Restart ON to OFF: Restart completed Turned OFF by the system when the Unit has been restarted.	Retained	Cleared	A302.00 to A302.15
A502 to A507	A502.00 to A507.15	Special I/O Unit Restart Bits	Turn these bits ON to restart (initialize) the Special I/O Unit with the corresponding unit number. Bits A502.00 to A507.15 correspond to unit numbers 0 to 95. When a restart bit is turned ON, the corresponding Special I/O Unit Initializing Flag (A330.00 to A335.15) will be turned ON. Both the restart bit and initializing flag will be turned OFF automatically when initialization is completed.	OFF to ON: Restart ON to OFF: Restart completed Turned OFF by the system when the Unit has been restarted.	Retained	Cleared	A330.00 to A335.15
A508	A508.09	Differentiate Monitor Completed Flag	ON when the differentiate monitor condition has been established during execution of differentiation monitoring. This flag will be cleared to 0 when differentiation monitoring starts.	ON: Monitor condition established OFF: Not yet established	Retained	Cleared	---
	A508.11	Trace Trigger Monitor Flag	ON when a trigger condition is established by the Trace Start Bit (A508.14). OFF when the next Data Trace is started by the Sampling Start bit (A508.15).	ON: Trigger condition established OFF: Not yet established or not tracing	Retained	Cleared	---
	A508.12	Trace Completed Flag	ON when sampling of a region of trace memory has been completed during execution of a Trace. OFF when the next time the Sampling Start Bit (A508.15) is turned from OFF to ON.	ON: Trace completed OFF: Not tracing or trace in progress	Retained	Cleared	---

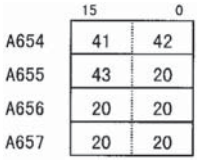
Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A508	A508.13	Trace Busy Flag	ON when the Sampling Start Bit (A508.15) is turned from OFF to ON. OFF when the trace is completed.	ON: Trace in progress OFF: Not tracing (not sampling)	Retained	Cleared	---
	A508.14	Trace Start Bit	Turn this bit from OFF to ON to establish the trigger condition. The offset indicated by the delay value (positive or negative) determines which data samples are valid.	ON: Trace trigger condition established OFF: Not established	Retained	Cleared	---
	A508.15	Sampling Start Bit	When a data trace is started by turning this bit from OFF to ON from the CX-Programmer, the PLC will begin storing data in Trace Memory by one of the three following methods: 1. Data is sampled at regular intervals (10 to 2,550 ms). 2. Data is sampled when TRSM(045) is executed in the program. 3. Data is sampled at the end of every cycle. The operation of A50815 can be controlled only from the CX-Programmer.	OFF to ON: Starts data trace (sampling) Turned ON from Programming Device.	Retained	Cleared	---
A510 and A511	---	Startup Time	These words contain the time at which the power was turned ON. The contents are updated every time that the power is turned ON. The data is stored in BCD. A510.00 to A510.07: Second (00 to 59) A510.08 to A510.15: Minute (00 to 59) A511.00 to A511.07: Hour (00 to 23) A511.08 to A511.15: Day of month (01 to 31)	See Function column.	Retained	See Function column.	Refreshed when power is turned ON.
A512 and A513	---	Power Interruption Time	These words contain the time at which the power was interrupted. The contents are updated every time that the power is interrupted. The data is stored in BCD. A512.00 to A512.07: Second (00 to 59) A512.08 to A512.15: Minute (00 to 59) A513.00 to A513.07: Hour (00 to 23) A513.08 to A513.15: Day of month (01 to 31) These words are not cleared at startup.	See Function column.	Retained	Retained	Written at power interruption
A514	---	Number of Power Interruptions	Contains the number of times that power has been interrupted since the power was first turned ON. The data is stored in binary. To reset this value, overwrite the current value with 0000 hex. This word is not cleared at startup, but it is cleared when the Memory Corruption Detected Flag (A395.11) goes ON.	0000 to FFFF hexadecimal	Retained	Retained	Refreshed when power is turned ON. A395.11
A515 to A517	---	Operation Start Time	The time that operation started as a result of changing the operating mode to RUN or MONITOR mode is stored here in BCD. A515.00 to A515.07: Seconds (00 to 59) A515.08 to A515.15: Minutes (00 to 59) A516.00 to A516.07: Hour (00 to 23) A516.08 to A516.15: Day of month (01 to 31) A517.00 to A517.07: Month (01 to 12) A517.08 to A517.15: Year (00 to 99) The previous start time is stored after turning ON the power supply until operation is started.	See Function column.	Retained	Retained	See Function column.

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A518 to A520	---	Operation End Time	The time that operation stopped as a result of changing the operating mode to PROGRAM mode is stored here in BCD. A518.00 to A518.07: Seconds (00 to 59) A518.08 to A518.15: Minutes (01 to 59) A519.00 to A519.07: Hour (00 to 23) A519.08 to A519.15: Day of month (01 to 31) A520.00 to A520.07: Month (01 to 12) A520.08 to A520.15: Year (00 to 99) If an error occurs in operation, the time of the error will be stored. If the operating mode is then changed to PROGRAM mode, the time that PROGRAM mode was entered will be stored.	See Function column.	Retained	Retained	See Function column.
A523	---	Total Power ON Time	Contains the total time that the PLC has been on in 10-hour units. The data is stored in binary and it is updated every 10 hours. To reset this value, overwrite the current value with 0000 hex. This word is not cleared at startup, but it is cleared to 0000 when the Memory Corruption Detected Flag (A395.11) goes ON.	0000 to FFFF hexadecimal	Retained	Retained	---
A526	A526.00	Serial Port Restart Bit	Turn this bit ON to restart the RS-232C port. (Do not use this bit when the port is operating in Peripheral Bus Mode.) This bit is turned OFF automatically when the restart processing is completed.	OFF to ON: Restart	Retained	Cleared	---
A527	A527.00 to A527.07	Online Editing Disable Bit Validator	The Online Editing Disable Bit (A527.09) is valid only when this byte contains 5A. To disable online editing from the CX-Programmer, set this byte to 5A and turn ON A527.09. (Online editing refers to changing or adding to the program while the PLC is operating in MONITOR mode.)	5 A: A527.09 enabled Other value: A527.09 disabled	Retained	Cleared	A527.09
	A527.09	Online Editing Disable Bit	Turn this bit ON to disable online editing. The setting of this bit is valid only when A527.00 to A527.07 have been set to 5 A.	ON: Disabled OFF: Not disabled	Retained	Cleared	A527.00 to A527.07
A528	A528.00 to A528.07	Serial Port Error Flags	These flags indicate what kind of error has occurred at the RS-232C port; they are automatically turned OFF when the RS-232C port is restarted. (These flags are valid in Serial Gateway mode. They are not valid in peripheral bus mode and only bit 5 is valid in NT Link mode.) Only the following bits are valid in Serial PLC Link Mode. Polling Unit: Bit 5: ON for timeout error. Polled Units: Bit 3: ON for framing error. Bit 4: ON for overrun error. Bit 5: ON for timeout error. These bits can be cleared by the CX-Programmer.	Bits 0 and 1: Not used. Bit 2: ON for parity error. Bit 3: ON for framing error. Bit 4: ON for overrun error. Bit 5: ON for timeout error. Bits 6 and 7: Not used.	Retained	Cleared	---
A529	---	FAL/FALS Number for System Error Simulation	Set a dummy FAL/FALS number to use to simulate the system error using FAL(006) or FALS(007). When FAL(006) or FALS(007) is executed and the number in A529 is the same as the one specified in the operand of the instruction, the system error given in the operand of the instruction will be generated instead of a user-defined error.	0001 to 01FF hex: FAL/FALS numbers 1 to 511 0000 or 0200 to FFFF hex: No FAL/FALS number for system error simulation. (No error will be generated.)	Retained	Cleared	---

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A530	---	Power Interrupt Disabled Area Setting	Set to A5A5 hex to disable power interrupts (except the Power OFF Interrupt task) between DI(693) and EI(694) instructions.	A5A5 hex: Masking power interruption processing enabled Other: Masking power interruption processing not enabled.	Cleared	Cleared	---
A580	A580.00 to A580.03	FB Communications Instruction Retries	Automatically stores the number of retries in the FB communications instruction settings specified in the PLC Setup. These Auxiliary Area bits/words are not to be written by the user. The number of resends and response monitoring time must be set by the user in the FB communications instructions settings in the PLC Setup, particularly when using function blocks from the OMRON FB Library to execute FINS messages or DeviceNet explicit messages communications. The values set in the Settings for OMRON FB Library in the PLC Setup will be automatically stored in the related Auxiliary Area words A580 to A582 and used by the function blocks from the OMRON FB Library.	0 to F hex	As set in PLC Setup	Cleared	Written at start of operation
A581	---	FB Communications Instruction Response Monitoring Time	Automatically stores the FB communications instruction response monitoring time set in the PLC Setup. These Auxiliary Area bits/words are not to be written by the user. The number of resends and response monitoring time must be set by the user in the FB communications instructions settings in the PLC Setup, particularly when using function blocks from the OMRON FB Library to execute FINS messages or DeviceNet explicit messages communications. The values set in the Settings for OMRON FB Library in the PLC Setup will be automatically stored in the related Auxiliary Area words A580 to A582 and used by the function blocks from the OMRON FB Library.	0001 to FFFF hex (Unit: 0.1 s; Range: 0.1 to 6553.5) 0000 hex: 2 s	As set in PLC Setup	Cleared	Written at start of operation
A582	---	FB DeviceNet Communications Instruction Response Monitoring Time	Automatically stores the FB DeviceNet communications instruction response monitoring time set in the PLC Setup. When FAL(006) or FALS(007) is executed and the number in A529 is the same as the one specified in the operand of the instruction, the system error given in the operand of the instruction will be generated instead of a user-defined error.	0001 to FFFF hex (Unit: 0.1 s; Range: 0.1 to 6553.5) 0000 hex: 2 s	As set in PLC Setup	Cleared	Written at start of operation
A595 and A596	---	IR00 Output for Background Execution	When an index register is specified as the output for an instruction processed in the background, A595 and A596 receive the output instead of IR00.	0000 0000 to FFFF FFFF hex (A596 contains the leftmost digits.)	Cleared	Cleared	---
A597	---	DR00 Output for Background Execution	When a data register is specified as the output for an instruction processed in the background, A597 receives the output instead of DR00.	0000 to FFFF hex	Cleared	Cleared	---
A598	A598.00	FPD Teaching Bit	Turn this bit ON to set the monitoring time automatically with the teaching function. While A598.00 is ON, FPD(269) measures how long it takes for the diagnostic output to go ON after the execution condition goes ON. If the measured time exceeds the monitoring time, the measured time is multiplied by 1.5 and that value is stored as the new monitoring time. The teaching function can be used only when a word address has been specified for the monitoring time operand.	ON: Teach monitoring time OFF: Teaching function off	Cleared	Cleared	---
	A598.01	Equals Flag for Background Execution	Turns ON if matching data is found for an SRCH(181) instruction executed in the background.	ON: Search data found in table OFF: Search data not found	Cleared	Cleared	---

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A600 to A603	---	Macro Area Input Words	When MCRO(099) is executed, the contents of the four specified source words (starting from the 1st input parameter word) are copied here. The contents of the four copied words are then passed to the subroutine.	Input data: 4 words	Cleared	Cleared	---
A604 to A607	---	Macro Area Output Words	After the subroutine specified in MCRO(099) has been executed, the results of the subroutine are transferred from A604 through A607 to the specified destination words. (output parameter words)	Output data: 4 words	Cleared	Cleared	---
A619	A619.02	Serial Port Settings Changing Flag	ON while the RS-232C port's communications settings are being changed. This flag will be turned ON when STUP(237) is executed and it will be turned OFF after the settings have been changed.	ON: Changing OFF: Not changing	Retained	Cleared	---

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A620	A620.01	Communications Unit 0, Port 1 Settings Changing Flag	The corresponding flag will be ON when the settings for that port are being changed. The flag will be turned ON when STUP(237) is executed and it will be turned OFF by an event issued from the Serial Communications Unit after the settings have been changed. It is also possible for the user to indicate a change in serial port settings by turning these flags ON.	ON: Changing OFF: Not changing	Retained	Cleared	---
	A620.02	Communications Unit 0, Port 2 Settings Changing Flag					---
	A620.03	Communications Unit 0, Port 3 Settings Changing Flag					---
	A620.04	Communications Unit 0, Port 4 Settings Changing Flag					---
A621 to A635	A621.00 to A635.04	Communications Units 1 to 15, Ports 1 to 4 Settings Changing Flag					---
A650	A650.00 to A650.07	Program Replacement Completed Flag	• Normal completion (i.e., when A650.14 is OFF) 01 hex: The program file (.OBJ) was replaced. • Error (i.e., when A650.14 is ON) 00 hex: A fatal error occurred. 01 hex: A memory error occurred. 11 hex: Memory is write-protected. 12 hex: The program replacement password is wrong. 21 hex: There is no Memory Card. 22 hex: The specified file does not exist. 23 hex: The size of the specified file exceeds the capacity for the model (memory error). 31 hex: One of the following operations is being executed. • An operation related to file memory is being executed. • A user program is being written. • The operating mode is being changed.	---	Retained	Cleared	---
	A650.14	Replacement Error Flag	ON when the Replacement Start Bit (A650.15) is turned ON to replace the program, but there is an error. If the Replacement Start Bit is turned ON again, the Replacement Error Flag will be turned OFF.	ON: Replacement error OFF: No replacement error, or the Replacement Start Bit (A65015) is ON.	Retained	Cleared	---
	A650.15	Replacement Start Bit	Program replacement starts when the Replacement Start Bit is turned ON if the Program Password (A651) is valid (A5A5 hex). Do not turn OFF the Replacement Start Bit during program replacement. When the power is turned ON or program replacement is completed, the Replacement Start Bit will be turned OFF, regardless of whether replacement was completed normally or in error. It is possible to confirm if program replacement is being executed by reading the Replacement Start Bit using the CX-Programmer, PT, or host computer.	ON: Program replaced OFF: Replacement completed, or after power is turned ON	Retained	Cleared	---

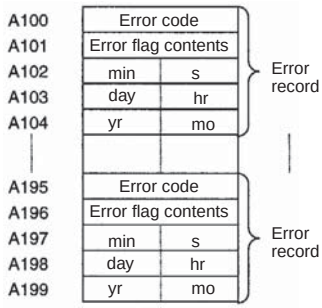
Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A651	---	Program Password	Type in the password to replace a program. A5A5 hex: Replacement Start Bit (A650.15) is enabled. Any other value: Replacement Start Bit (A650.15) is disabled. When the power is turned ON or program replacement is completed, the Replacement Start Bit will be turned OFF, regardless of whether replacement was completed normally or in error.	---	Retained	Cleared	---
A654 to 657	---	Program File Name	When program replacement starts, the program file name will be stored in ASCII. File names can be specified up to eight characters in length excluding the extension. File names are stored in the following order: A654 to A657 (i.e., from the lowest word to the highest), and from the highest byte to the lowest. If a file name is less than eight characters, the lowest remaining bytes and the highest remaining word will be filled with spaces (20 hex). Null characters and space characters cannot be used within file names. Example: File name is ABC.OBJ  	---	Retained	Cleared	---
A720 to A722	---	Power ON Clock Data 1	These words contain the same time data as the startup time stored in words A510 to A511, as well as the month and year information. A720.00 to A720.07: Seconds (00 to 59) A720.08 to A720.15: Minutes (00 to 59) A721.00 to A721.07: Hour (00 to 23) A721.08 to A721.15: Day of month (01 to 31) A722.00 to A722.07: Month (01 to 12) A722.08 to A722.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A723 to A725	---	Power ON Clock Data 2	These words contain the time at which the power was turned ON one time before the startup time stored in words A510 and A511. A723.00 to A723.07: Seconds (00 to 59) A723.08 to A723.15: Minutes (00 to 59) A724.00 to A724.07: Hour (00 to 23) A724.08 to A724.15: Day of month (01 to 31) A725.00 to A725.07: Month (01 to 12) A725.08 to A725.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A726 to A728	---	Power ON Clock Data 3	These words contain the time at which the power was turned ON two times before the startup time stored in words A510 and A511. A726.00 to A726.07: Seconds (00 to 59) A726.08 to A726.15: Minutes (00 to 59) A727.00 to A727.07: Hour (00 to 23) A727.08 to A727.15: Day of month (01 to 31) A728.00 to A728.07: Month (01 to 12) A728.08 to A728.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A729 to A731	---	Power ON Clock Data 4	These words contain the time at which the power was turned ON three times before the startup time stored in words A510 and A511. A729.00 to A729.07: Seconds (00 to 59) A729.08 to A729.15: Minutes (00 to 59) A730.00 to A730.07: Hour (00 to 23) A730.08 to A730.15: Day of month (01 to 31) A731.00 to A731.07: Month (01 to 12) A731.08 to A731.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A732 to A734	---	Power ON Clock Data 5	These words contain the time at which the power was turned ON four times before the startup time stored in words A510 and A511. A732.00 to A732.07: Seconds (00 to 59) A732.08 to A732.15: Minutes (00 to 59) A733.00 to A733.07: Hour (00 to 23) A733.08 to A733.15: Day of month (01 to 31) A734.00 to A734.07: Month (01 to 12) A734.08 to A734.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A735 to A737	---	Power ON Clock Data 6	These words contain the time at which the power was turned ON five times before the startup time stored in words A510 and A511. A735.00 to A735.07: Seconds (00 to 59) A735.08 to A735.15: Minutes (00 to 59) A736.00 to A736.07: Hour (00 to 23) A736.08 to A736.15: Day of month (01 to 31) A737.00 to A737.07: Month (01 to 12) A737.08 to A737.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A738 to A740	---	Power ON Clock Data 7	These words contain the time at which the power was turned ON six times before the startup time stored in words A510 and A511. A738.00 to A738.07: Seconds (00 to 59) A738.08 to A738.15: Minutes (00 to 59) A739.00 to A739.07: Hour (00 to 23) A739.08 to A739.15: Day of month (01 to 31) A740.00 to A740.07: Month (01 to 12) A740.08 to A740.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A741 to A743	---	Power ON Clock Data 8	These words contain the time at which the power was turned ON seven times before the startup time stored in words A510 and A511. A741.00 to A741.07: Seconds (00 to 59) A741.08 to A741.15: Minutes (00 to 59) A742.00 to A742.07: Hour (00 to 23) A742.08 to A742.15: Day of month (01 to 31) A743.00 to A743.07: Month (01 to 12) A743.08 to A743.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.
A744 to A746	---	Power ON Clock Data 9	These words contain the time at which the power was turned ON eight times before the startup time stored in words A510 and A511. A744.00 to A744.07: Seconds (00 to 59) A744.08 to A744.15: Minutes (00 to 59) A745.00 to A745.07: Hour (00 to 23) A745.08 to A745.15: Day of month (01 to 31) A746.00 to A746.07: Month (01 to 12) A746.08 to A746.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.

Address		Name	Function	Settings	Status after mode change	Status at startup	Write timing/ Related flags, settings
Words	Bits						
A747 to A749	---	Power ON Clock Data 10	These words contain the time at which the power was turned ON nine times before the startup time stored in words A510 and A511. A747.00 to A747.07: Seconds (00 to 59) A747.08 to A747.15: Minutes (00 to 59) A748.00 to A748.07: Hour (00 to 23) A748.08 to A748.15: Day of month (01 to 31) A749.00 to A749.07: Month (01 to 12) A749.08 to A749.15: Year (00 to 99)	See Function column.	Retained	Retained	Written when power is turned ON.

A-3-3 Details on Auxiliary Area Operation

A100 to A199: Error Log Area



The following data would be generated in an error record if a memory error (error code 80F1) occurred on 1 April 1998 at 17:10:30 with the error located in the PLC Setup (04 hex).

80	F1
00	04
10	30
01	17
98	04

The following data would be generated in an error record if an FALS error with FALS number 001 occurred on 2 May 1997 at 8:30:15.

C1	01
00	00
30	15
02	08
97	05

Error Codes and Error Flags

Data type	Error code	Error name	Error flags
Fatal errors from the system	0x809F	Cycle Time Exceeded Error	---
	80C0 to 80C7, 80CE, 80CF	I/O Bus Error	A404
	80E0	I/O Setting Error	---
	80E1	Too Many I/O Points Error	A407
	80E9	Unit Number Duplication Error	A410, A411 to A416*1
	80EA	Rack Number Duplication Error	A409
	80F0	Program Error	A295 to A299*2
	80F1	Memory Error	A403
	80F2	Version Error	---
	80F4	Memory Card Transfer Error	---
User-defined fatal errors	C101 to C2FF	FALS instruction executed*3	---
Non-fatal errors from the system	008B	Duplicate Refresh Error	A426
	009A	Basic I/O Unit Error	A408
	009B	PLC Setup Error	A406
	00D1	Option Board Error	---
	00F1	Backup Memory Error	A315.15
	00F2	Tag Memory Error*4	A315.12
	00F3	System Work Memory Error	A402.00
	00F7	Battery Error	---
	0200 to 020F	CPU Bus Unit Error	A417
	0300 to 035F	Special I/O Unit Error	A418 to A423*5
	0400 to 040F	CPU Bus Unit Setup Error	A427
	0500 to 055F	Special I/O Unit Setup Error	A428 to A433*5
User-defined non-fatal errors	4101 to 42FF	FAL instruction executed*6	---

Note 1 The contents of the error flags for a number duplication error are as follows:
Bits 0 to 7: Unit number (binary), 00 to 5F hex for Special I/O Units, 00 to 0F hex for CPU Bus Units
Bits 8 to 14: All zeros.
Bit 15: Unit type OFF: CPU Bus Unit
ON: Special I/O Unit

2 Only the contents of A295 is stored as the error flag contents for program errors.

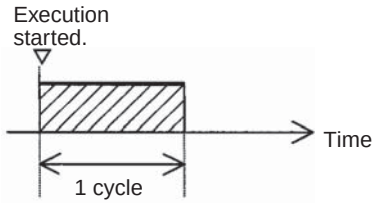
3 C101 to C2FF will be stored for FALS numbers 001 to 511.

4 Supported only by the CJ2H-CPU6□-EIP and CJ2M-CPU3□.

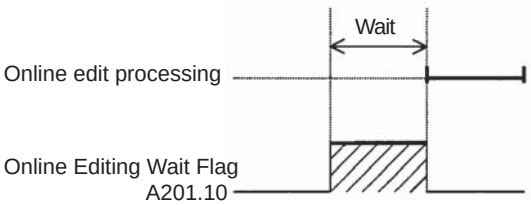
5 A value of 0000 hex will be stored as the error flag contents.

6 A value of 4101 to 42FF will be stored for FAL numbers 001 to 511.

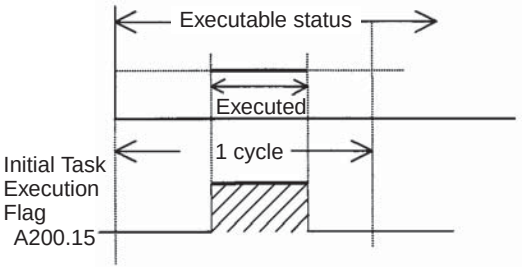
A200.11: First Cycle Flag



A201.10: Online Editing Wait Flag

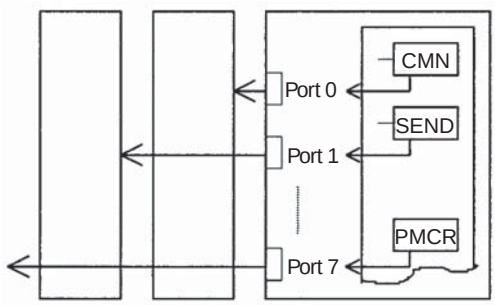


A200.15: Initial Task Flag

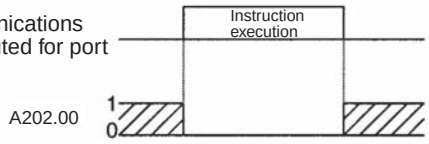


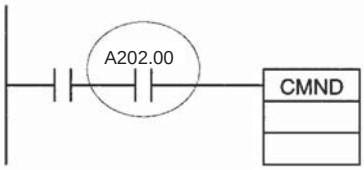
A200.15 will turn ON during the first time a task is executed after it has reached executable status. It will be ON only while the task is being executed and will not turn ON if following cycles.

A202.00 to A202.07: Communications Port Enabled Flags



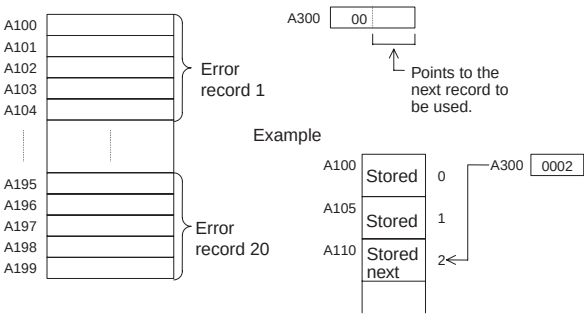
Network communications instruction executed for port 0.



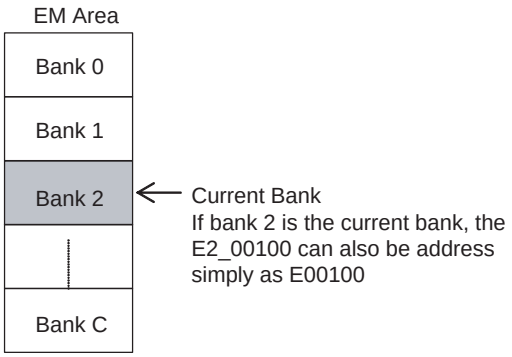


The program is designed so that CMND(490) will be executed only when A202.00 is ON.

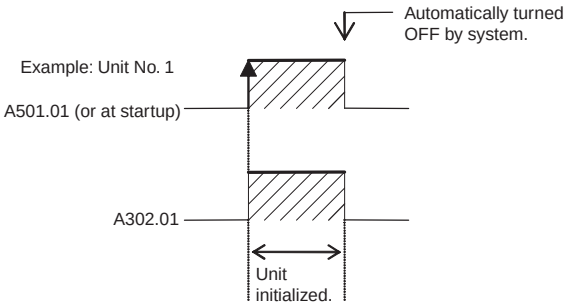
A300: Error Record Pointer



A301: Current EM Bank



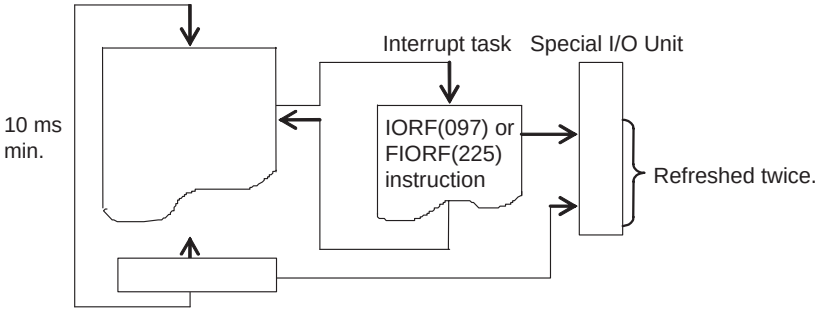
A501.01 to A501.15: CPU Bus Unit Restart Bits and



A401.09: Program Error

	Error	Address
Program Errors when A401.09 is ON	UM Overflow Error Flag	A295.15
	Illegal Instruction Flag	A295.14
	Distribution Overflow Error Flag	A295.13
	Task Error Flag	A259.12
	No END(001) Error Flag	A295.11
	Illegal Area Access Error Flag	A295.10
	Indirect DM/EM Addressing Error Flag	A295.09

A426.15: Duplicate Refresh Error Cause



A-4 Memory Map of PLC Memory Addresses

A-4-1 PLC Memory Addresses

PLC memory addresses are set in Index Registers (IR00 to IR15) to indirectly address I/O memory. Normally, use the MOVE TO REGISTER (MOVR(560)) and MOVE TIMER/COUNTER PV TO REGISTER (MOVRW(561)) instructions to set PLC memory addresses into the Index Registers.

Some instructions, such as DATA SEARCH (SRCH(181)), FIND MAXIMUM (MAX(182)), and FIND MINIMUM (MIN(183)), output the results of processing to an Index Register to indicate an PLC memory address.

There are also instructions for which Index Registers can be directly designated to use the PLC memory addresses stored in them by other instructions. These instructions include DOUBLE MOVE (MOVL(498)), some symbol comparison instructions (=L,<>L, <L, >L,<=L, and >=L), DOUBLE COMPARE (CMPL(060)), DOUBLE DATA EXCHANGE (XCGL(562)), DOUBLE INCREMENT BINARY (++L(591)), DOUBLE DECREMENT BINARY (--L(593)), DOUBLE SIGNED BINARY ADD WITHOUT CARRY (+L(401)), DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY (-L(411)), SET RECORD LOCATION (SETR(635)), and GET RECORD LOCATION (GETR(636)).

The PLC memory addresses all are continuous and the user must be aware of the order and boundaries of the memory areas. As reference, the PLC memory addresses are provided in a table at the end of this appendix.



Precautions for Correct Use

Directly setting PLC memory addresses in the program should be avoided whenever possible. If PLC memory addresses are set in the program, the program will be less compatible with new CPU Unit models or CPU Units for which changed have been made to the layout of the memory.



Precautions for Correct Use

Address Offset and Array Variable Precautions

When using the following functions, a PLC memory address in the next data area may be specified depending on the offset value.

- When using an offset to indirectly specify an address.
- When indirectly specifying an element number in an array variable.

Be sure to program so that the data area is not exceeded, or consider the distribution order of the data area of this memory map when programming.

A-4-2 Memory Map

The following table shows the PLC memory addresses of the I/O memory area. The I/O memory area is backed up by a battery. If there is no battery, the contents of the I/O memory area may become corrupted.

Note Do not access the areas indicated *Reserved for system*.

Classification	PLC memory addresses (hex)	User addresses	Area
I/O memory areas	00000 to 0B7FF		Reserved for system.
	0B800 to 0B801	TK0000 to TK0031	Task Flag Area
	0B802 to 0B83F		Reserved for system.
	0B840 to 0B9FF	A000 to A447	Read-only Auxiliary Area
	0BA00 to 0BDFF	A448 to A1471	Read/Write Auxiliary Area
		A10000 to A11535	Read-only Auxiliary Area
	0BE00 to 0BEFF	T0000 to T4095	Timer Completion Flags
	0BF00 to 0BFFF	C0000 to T4095	Counter Completion Flags
	0C000 to 0D7FF	CIO 0000 to CIO 6143	CIO Area
	0D800 to 0D9FF	H000 to H511	Holding Area
	0DA00 to 0DDFF	H512 to H1535	Holding Area These words are used for function blocks only.
	0DE00 to 0DFFF	W000 to W511	Work Area
	0E000 to 0EFFF	T0000 to T4095	Timer PVs
	0F000 to 0FFFF	C0000 to C4095	Counter PVs
	10000 to 17FFF	D00000 to D32767	DM Area
	18000 to 1FFFF	E0_00000 to E0_32767	EM Area bank 0
	20000 to 27FFF	E1_00000 to E1_32767	EM Area bank 1 hex
	:	:	:
	D8000 to DFFFF	E18_00000 to E18_32767	EM Area bank 18 hex
	:	:	
	F8000 to FFFFF	E_00000 to E_32767	EM Area, current bank (See note.)

Note The contents of the EM Area bank currently specified in the program is stored at these addresses. For example, if bank 1 is specified, the same contents as at 20000 to 27FFF will be stored at F8000 to FFFFF.

A-5 Operation for Power Interruptions

A-5-1 Power OFF Operation

The following processing is performed if CPU Unit power is turned OFF. Power OFF processing will be performed if the power supply falls below 85% (80% for CJ1W-PD025 DC Power Supply Units or 90% for CJ1W-PD022 DC Power Supply Units) of the minimum rated voltage while the CPU Unit is in RUN or MONITOR mode.

1 The CPU Unit will stop.

2 All outputs from Output Units will be turned OFF.

Note 1 All outputs will turn OFF regardless of the status of the IOM Hold Bit or the setting of the IOM Hold Bit at startup in the PLC Setup.

2 85% of the rated voltage for AC Power Supply Unit:
100 VAC:85 VAC
200 VAC:170 VAC
100 to 240-V input (wide range): 85 VAC

3 80%/90% of the rated voltage for DC Power Supply Unit:
CJ1W-PD025: 19.2 VDC
CJ1W-PD022: 21.6 VDC

Momentary Power Interruptions

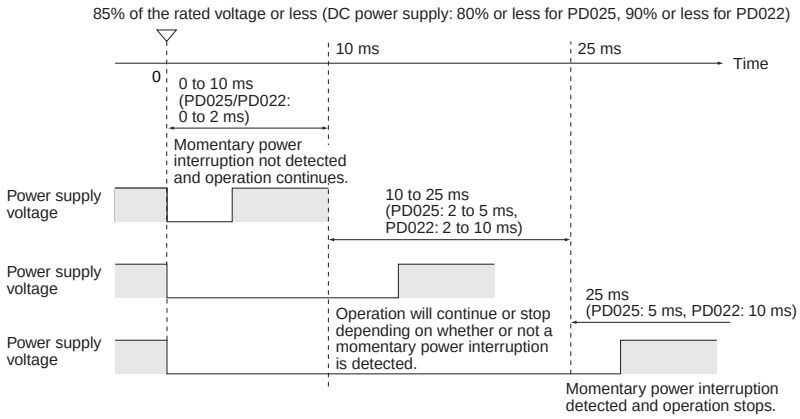
The following processing will be performed if power drops only momentarily (momentary power interruption).

- The system will continue to run unconditionally if the momentary power interruption lasts less than 10 ms, i.e., the time it takes the minimum rated voltage at 85% or less to return to 85% or higher is less than 10 ms.

Note When DC power supplies are used, less than 2 ms is required for the rated voltage of a CJ1W-PD025 at 80% or less to return to 80% or higher or for the rated voltage of a CJ1W-PD022 at 90% or less to return to 90% or higher.

- A momentary power interruption that lasts more than 10 ms but less than 25 ms (when using a DC power supply, more than 2 ms but less than 5 ms for CJ1W-PD025 and more than 2 ms but less than 10 ms for CJ1W-PD022) is difficult to determine and a power interruption may or may not be detected.
- The system will stop unconditionally if the momentary power interruption lasts 25 ms or longer (when using a DC power supply, 5 ms or longer for CJ1W-PD025 and 10 ms or longer for CJ1W-PD022).

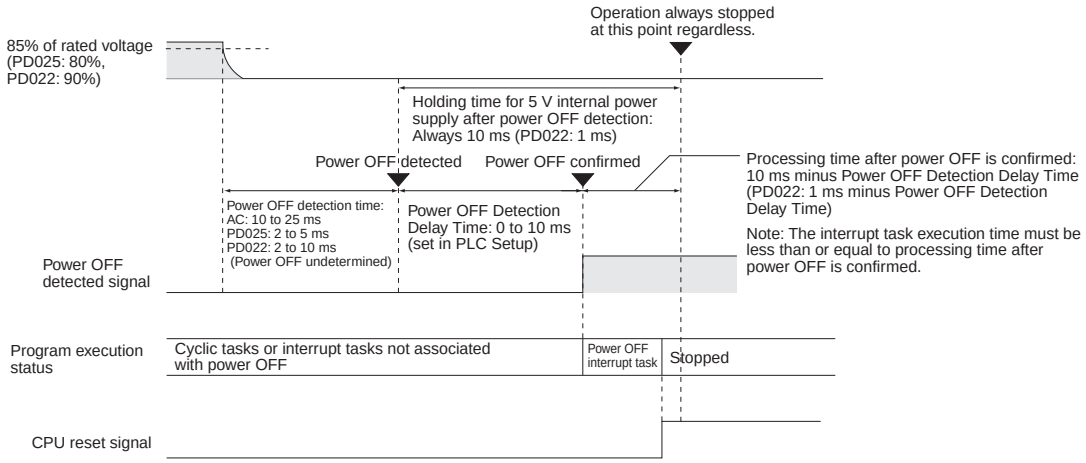
Therefore, the time required to detect a power interruption is 10 to 25 ms (when using a DC power supply, 2 to 5 ms for CJ1W-PD025 and 2 to 10 ms for CJ1W-PD022). If operation stops under the conditions given in items 2 and 3 above, the timing used to stop operation (or the timing used to start execution of the Power OFF Interrupt Task) can be delayed by setting the Power OFF Detection Delay Time (0 to 10 ms) in the PLC Setup. The holding time for the 5-VDC outputs of CJ1W-PD022 Power Supply Units when a power interruption occurs is 1 ms, however, so the Power OFF Interrupt Task Disable and Power OFF Detection Time parameters cannot be used.



Note The above timing chart shows an example when the power OFF detection delay time is set to 0 ms.

Power OFF Timing Chart

The following timing chart shows the CPU Unit power OFF operation in more detail.



Power OFF Detection Time:	The time it takes to detect power OFF after the power supply falls below 85% (80% for CJ1W-PD025 DC Power Supply Units or 90% for CJ1W-PD022 DC Power Supply Units) of the minimum rated voltage.
Power OFF Detection Delay Time:	The delay time after power OFF is detected until it is confirmed. This can be set in the PLC Setup within a range from 0 to 10 ms. If an unstable power supply is causing power interruptions, set a longer Power OFF Detection Delay Time (10 ms max.) in the PLC Setup. The CJ1W-PD022 Power Supply Units only support a holding time of 1 ms, however, so this setting is not possible.
Power Holding Time:	The maximum amount of time (fixed at 10 ms) that 5 V will be held internally after power shuts OFF. The time that it takes for the power OFF interrupt task to execute must not exceed 10 ms minus the Power OFF Detection Delay Time (processing time after power OFF is confirmed). The power OFF interrupt task will be ended even if it has not been completely executed the moment this time expires. The CJ1W-PD022 Power Supply Units only support a holding time of 1 ms, however, so this setting is not possible.

Description of Operation

- Power OFF will be detected if the 100 to 240-VDC power supply falls below 85% (80% for CJ1W-PD025 DC Power Supply Units or 90% for CJ1W-PD022 DC Power Supply Units) of the minimum rated voltage for the power OFF detection time (somewhere between 10 to 25 ms for AC Power Supply Units, somewhere between 2 to 5 ms for CJ1W-PD025 DC Power Supply Units, and somewhere between 2 to 10 ms for CJ1W-PD022 DC Power Supply Units).

- If the Power OFF Detection Delay Time is set (0 to 10 ms) in the PLC Setup, then the following operations will be performed when the set time expires.
  - If the power OFF interrupt task is disabled (default PLC Setup setting)
    The CPU reset signal will turn ON and the CPU will be reset immediately.
  - If the power OFF interrupt task is enabled (in the PLC Setup), the CPU reset signal will turn ON and the CPU will be reset after the power OFF interrupt task has been executed. Make sure that the power OFF interrupt task will finish executing within 10 ms minus the Power OFF Detection Delay Time = processing time after power OFF. The 5-V internal power supply will be maintained only for 10 ms after power OFF is detected. The holding time for the internal 5-V power supply of CJ1W-PD022 Power Supply Units when a power interruption occurs is 1 ms, however, so the Power OFF Detection Delay Time and Power OFF Interrupt Task Disable functions cannot be used.

A-5-2 Instruction Execution for Power Interruptions

If power is interrupted and the interruption is confirmed when the CPU Unit is operating in RUN or MONITOR mode, the instruction currently being executed will be completed*1 and the following power interruption processing will be performed.

- If the power OFF interrupt task has not been enabled, the CPU Unit will be reset immediately.
- If the power OFF interrupt task has been enabled, the task will be executed and then the CPU Unit will be reset immediately.

The power OFF interrupt task is enabled and disabled in the PLC Setup.

*1 The current instruction can be completed only when the time required to complete execution is less than or equal to the processing time after power interruption detection (10 ms – power interruption detection delay time). If the instruction is not completed within this time, it will be interrupted and the above processing will be performed.

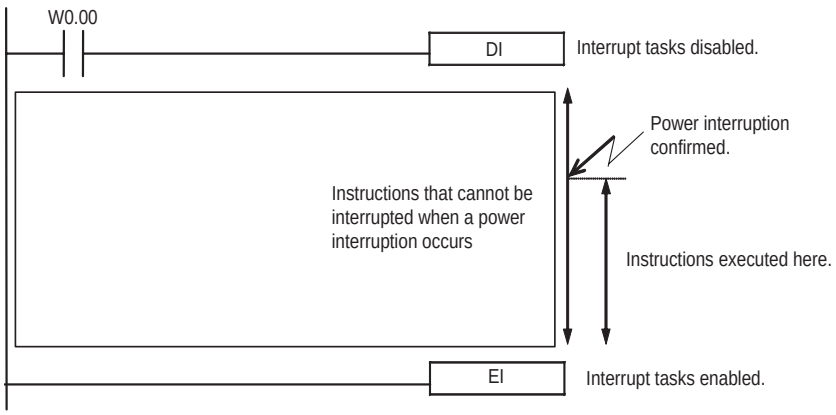
Note The processing time after a power interruption is detected is 1 ms when a CJ1W-PD022 is mounted.

Disabling Power Interruption Processing in the Program

If the power OFF interrupt task is disabled, areas of the program can be protected from power interruptions so that the instructions will be executed before the CPU Unit performs power OFF processing even if the power supply is interrupted. This is achieved by using the DISABLE INTERRUPTS (DI(693)) and ENABLE INTERRUPTS (EI(694)) instructions.

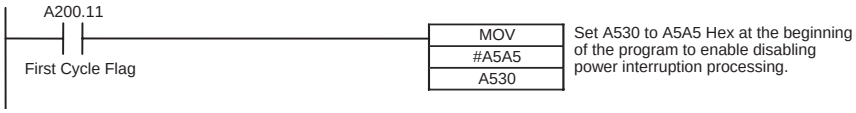
The following procedure is used.

1 Insert DI(693) before the program section to be protected to disable interrupts and then place EI(694) after the section to enable interrupts.



2 Set the Disable Setting for Power OFF Interrupts in A530 to A5A5 Hex to enable disabling power interruption processing.

Note A530 is normally cleared when power is turned OFF. To prevent this, the IOM Hold Bit (A500.12) must be turned ON and the PLC Setup must be set to maintain the setting of the IOM Hold Bit at Startup, or the following type of instruction must be included at the beginning of the program to set A530 to A5A5 Hex.



3 Disable the Power OFF Interrupt Task in the PLC Setup.

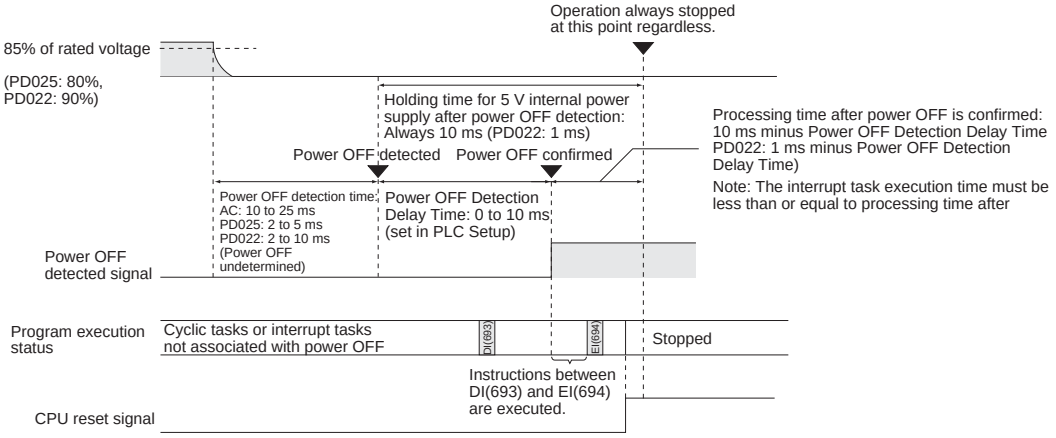
With the above procedure, all instructions between DI(693) and EI(694) (or END) will be completed*1 before the Power OFF Interrupt is executed even if the power interruption occurs while executing the instructions between DI(693) and EI(694).

*1 The protected instructions can be completed only when the time required to complete execution is less than or equal to the processing time after power interruption detection (10 ms – power interruption detection delay time). If the instructions is not completed within this time, they will be interrupted and the above processing will be performed.

Note 1 The processing time after a power interruption is detected is 1 ms when a CJ1W-PD022 is mounted.

2 If the Power OFF Interrupt Task is not disabled in the PLC Setup, the Power OFF Interrupt Task will be executed, and the CPU Unit will be reset without executing the protected instructions as soon as the power interruption is detected.

3 If a power interrupt is detected while DI(693) is being executed, the CPU Unit will be reset without executing the protected instructions.



If A530 (Power Interrupt Disabled Area Setting) is set to any value other than A5A5 hex to enable masking power interruption processing, power interrupt processing will be started after completing execution of the current instruction.

Additional Information

Interrupt processing is performed according to the contents of A530 and the PLC Setup as shown below.

Power interrupt processing		A530 (Power Interrupt Disabled Area Setting)	
		A5A5 hex (enable masking power interrupt processing)	Other (disable masking power interrupt processing)
Power OFF Interrupt Task (PLC Setup)	Disabled	All instructions between DI(693) and EI(694) are executed and the CPU Unit is reset.	Execution of the current instruction is completed and the CPU Unit is reset.
	Enabled	Execution of the current instruction is completed, the Power OFF Interrupt Task is executed, and the CPU Unit is reset.	

A-6 EtherNet/IP Connections from Windows XP (SP2 or Higher) or Windows Vista

Better firewall security for Windows XP (SP2 or higher) and Windows Vista has increased the restrictions for data communications on Ethernet ports. When using an EtherNet/IP connection to a CPU Unit from an Ethernet port on a computer, you must change the settings of the Windows Firewall to enable using CX-Programmer communications.

Precautions for Correct Use

Windows Firewall is mainly designed to prevent inappropriate access from external devices (e.g., via the Internet). The changes to the Windows Firewall settings described in this document enable EtherNet/IP connections to be used by the CX-Programmer. If the same computer is being used on a company network or other network, confirm that the changes will not create security problems before proceeding with the changes. The changes described in this document are required only when you connect using EtherNet/IP through an Ethernet port. No changes are necessary if you are connecting through any other port, such as a USB port.

A-6-1 Changing Windows Firewall Settings

Windows XP

1 When you attempt to connect the CX-Programmer to a PLC on an EtherNet/IP network through an Ethernet port, the Windows Security Alert Dialog Box will be displayed.

2 Click the **Unblock** Button.

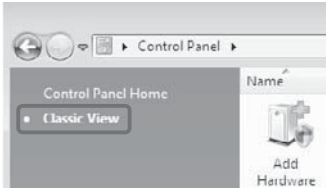


An EtherNet/IP connection will be accepted from CX-Programmer and EtherNet/IP connections will be enabled in the future as well.

Windows Vista

Use the following procedure to change the settings before attempting to connect from the CX-Programmer.
The User Account Control Dialog Box may be displayed during this procedure. If it appears, click the Continue Button and continue with the procedure.

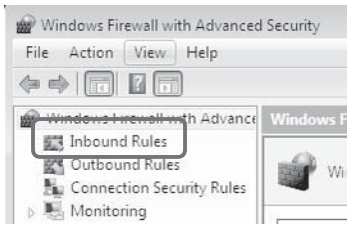
1 Select Control Panel from the Windows Start Menu and change the display to Classic View.



2 Open the Administrative Tools and select Windows Firewall with Advanced Security from the dialog box that is displayed.



3 Select Inbound Rules under Windows Firewall with Advanced Security on Local Computer on the left side of the Windows Firewall with Advanced Security Dialog Box.



4 Select New Rule under Inbound Rules in the Actions Area on the right side of the dialog box.



5 Make the following settings for each step in the New Inbound Rule Wizard Dialog Box, clicking the Next Button to move between steps.

Rule Type	Select *Custom*.
Program	Select *All Programs*.
Protocol and Ports	Select ICMPv4 as the protocol type. 
Scope	Select *Any IP address* for everything.
Action	Select *Allow the connection*.
Profile	Select *Domain*, *Private*, and *Public*.
Name	Enter any name, e.g., Omron_EIP.

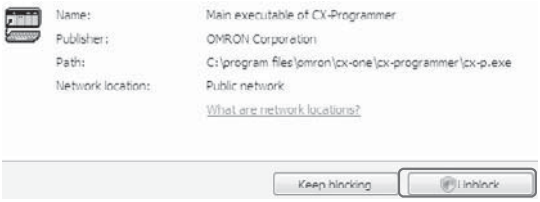
6 Click the **Finish** Button. The rule that you defined will be registered in the Inbound Rules (e.g., Omron_EIP).



7 Close the *Windows Firewall with Advanced Security* Dialog Box.

8 When you attempt to connect the CX-Programmer to a PLC on an EtherNet/IP network through an Ethernet port, the Windows Security Alert Dialog Box will be displayed.

9 Click the **Unblock** Button.



An EtherNet/IP connection will be accepted from CX-Programmer and EtherNet/IP connections will be enabled in the future as well.

A-7 PLC Comparison Charts: CJ-series and CS-series PLCs

The following table lists only the differences in functionality between the CJ2 CPU Units and other CS-series and CJ-series CPU Units.

Functional Comparison

Item		CJ Series					CS Series
		CJ2 CPU Units	CJ2M CPU Units	CJ1-H-R CPU Units	CJ1-H CPU Units	CJ1-M CPU Units	CS1-H CPU Units
General specifications	Dimensions (height × depth × width)	CJ2H-CPU6□-EIP: 90 × 65 × 80 mm CJ2H-CPU6□: 90 × 65 × 40 mm	CJ2H-CPU1□: 90 × 75 × 31 mm CJ2M-CPU3□: 90 × 75 × 62 mm	90 × 65 × 62 mm		90 × 65 × 31 mm 90 × 65 × 49 mm	130 × 101 × 71 mm
	Weight	CJ2H-CPU6□-EIP: 280 g max. CJ2H-CPU6□: 190 g max.	CJ2H-CPU1□: 130 g max. CJ2M-CPU3□: 190 g max. (without Serial Option Board)	200 g max.		120 g max. 170 g max.	350 g max.
	Current consumption at 5 VDC	CJ2H-CPU6□-EIP: 0.82 A CJ2H-CPU6□: 0.42 A	CJ2H-CPU1□: 0.5 A CJ2M-CPU3□: 0.7 A	0.99 A		0.58 A 0.64 A	0.82 A
	Vibration resistance	Conforms to JIS 60068-2-6.	Conforms to JIS C60068-2-6.	Conforms to JIS C0040.			
	Shock resistance	Conforms to JIS 60068-2-27.	Conforms to JIS C60068-2-27.	Conforms to JIS C0041.			
Performance specifications	Maximum program capacity	400K steps	60K steps	250K steps			
	Maximum I/O capacity	2,560 points					5,120 points
	System overhead time	CJ2H-CPU6□-EIP: 200 μs (including EtherNet/IP port overhead time) CJ2H-CPU6□: 100 μs	CJ2M-CPU3□: 270 μs CJ2M-CPU1□: 160 μs (If tag data links are used with EtherNet/IP, add the following to the above time: 100 μs + Number of transferred words × 1.8 μs.)	130 μs	300 μs	500 μs	300 μs
	Basic instruction execution time (LD)	0.016 μs	0.04 μs		0.02 μs	0.1 μs	0.02 μs
	Special instruction execution time (MOV)	0.048 μs	0.06 μs		0.18 μs	0.3 μs	0.18 μs
	Execution time for immediate update instructions (!LD)	0.99 μs	1.26 μs	21 μs		24 μs, 28 μs	21 μs
	Interrupt task start time	26 μs (30 μs for unit version 1.0)	31 μs	40 μs	124 μs	169 μs	124 μs
	Basic I/O refreshing (16-point Unit)	1.4 μs	3.9 μs		3 μs		
	Special I/O refreshing (8 analog inputs)	50 μs	70 μs		120 μs	160 μs	120 μs

Item		CJ Series					CS Series
		CJ2 CPU Units	CJ2M CPU Units	CJ1-H-R CPU Units	CJ1-H CPU Units	CJ1-M CPU Units	CS1-H CPU Units
Performance specifications	CPU Bus Unit refreshing (tag data links with EtherNet/IP Unit)	100 μs + (No. of words transferred × 0.33 μs)	100 μs + (No. of words transferred × 0.7 μs)	200 μs + (No. of words transferred × 0.7 μs)		270 μs + (No. of words transferred × 0.7 μs)	200 μs + (No. of words transferred × 0.7 μs)
	Maximum EM Area capacity	32K words × 25 banks	32K words × 4 banks	32K words × 13 banks			
	Source/comment memory	3.5 MB	1,024K bytes	2 MB			
	Function block definitions, actions, and transitions	2,048	CJ2M-CPU□1/CPU□2/CPU□3: 256 CJ2M-CPU□4/CPU□5: 2,048	1,024			
	Applicable Memory Cards	30, 64, 128, 256, or 512 MB	128 Mbytes 256 Mbytes 512 Mbytes	30, 64, or 128 MB			
	Number of cyclic tasks	128 tasks		32 tasks			
	Communications ports (internal logic ports)	8 + 64 ports		8 ports			
	Increase in cycle time for online editing	Approx. 1 ms		Approx. 8 ms		Approx. 14 ms	Approx. 8 ms.
	Maximum number of Expansion Racks	3 Racks					7 Racks
	Maximum number of Interrupt Input Units	2 Units					4 Units
	Maximum number of CPU Bus Units	15 Units	CJ2M-CPU1□: 16 Units CJ2M-CPU3□: 15 Units	16 Units			
	Inner Boards	Not applicable.					Applicable

Item		CJ Series					CS Series
		CJ2 CPU Units	CJ2M CPU Units	CJ1-H-R CPU Units	CJ1-H CPU Units	CJ1-M CPU Units	CS1-H CPU Units
Functional specifications	CPU processing mode	Normal processing mode		Normal mode or parallel processing mode			
	Synchronous unit operation	Supported (unit version 1.1 or later)	Not supported	Not supported			
	Special instructions for certain CPU Bus Units	Supported (unit version 1.1 or later)	Supported	Not supported			
	Minimum cycle time	0.2 to 32,000 ms in 0.1-ms increments The minimum cycle time can be changed during operation (unit version 1.1 or later).	0.2 to 32,000 ms in 0.1-ms increments The minimum cycle time can be changed during operation		1 to 32,000 ms in 1-ms increments		
	Unit for monitoring cycle time	0.01 ms		0.1 ms			
	Time unit for setting scheduled interrupts	0.1, 1, or 10 ms		1 or 10 ms		0.1, 1, or 10 ms	1 or 10 ms
	Communications ports	USB, RS-232C, and EtherNet/IP (CJ2H-CPU6□-EIP only)	USB, RS-232C, and RS-422A/485 (A Serial Option Board is required for the CJ2M-CPU3□.) EtherNet/IP (CJ2M-CPU3□ only)	Peripheral and RS-232C			
	Serial PLC Links	Not supported	Supported	Supported		Supported	Not supported
	Tag name server	CJ2H-CPU6□-EIP: Supported CJ2H-CPU6□: Not supported	CJ2M-CPU3□: Supported CJ2M-CPU1□: Not supported	Not provided			
	Slots for external interrupts	CJ2H-CPU6□-EIP: Slots 0 to 3 CJ2H-CPU6□: Slots 0 to 4	Slots 0 to 4	Slots 0 to 4			Slots 0 to 9
	Data tracing	32K words max., Sampling time: 1 to 2,550 ms, continuous tracing supported.	8K words max., Sampling time: 1 to 2,550 ms, continuous tracing supported.	4K words max., Sampling time: 10 to 2,550 ms			
	Write protection using lot number	Supported	Supported	Not supported			
	PLC names	Supported	Supported	Not supported			
	Addressing bits in DM and EM Areas	Supported	Supported	Not supported			
	Force-setting/resetting bits in EM Area	Supported with the EM Area force-set/reset function if the automatic address allocation is used or for unit version 1.2 or later.	Supported with EM Area force-set/reset function	Not supported			
	Converting part of EM Area to trace memory	Supported	Supported	Not supported			
	Specifying offsets for addresses	Supported	Supported	Not supported			
	Using BCD and binary refreshing for timer instructions in same program	Supported	Supported	Not supported			

Item		CJ Series					CS Series
		CJ2 CPU Units	CJ2M CPU Units	CJ1-H-R CPU Units	CJ1-H CPU Units	CJ1-M CPU Units	CS1-H CPU Units
Functional specifications	Number of entries in a relay network table (one of the routing tables)	128 max.			32 max.		
	Clock pulses	0.1 ms, 1 ms, 0.01 s, 0.02 s, 0.1 s, 0.2 s, 1 s, and 1 min			0.02 s, 0.1 s, 0.2 s, 1 s, and 1 min		
	Pulse I/O implementation method	Not supported by CPU Unit by itself. Only transistor NPN outputs.		Not supported by CPU Unit by itself. Only transistor NPN outputs.		Supported only by models with built-in I/O	Not supported by CPU Unit by itself. Only transistor NPN outputs.
	Normal inputs	Not supported		Not supported		10	Not supported
	Input interrupts	Not supported		Not supported		4 max. Min. input signal pulse width: 30 μs	Not supported
	Quick-response inputs	Not supported		Not supported		4 max. (total must include quick-response, high-speed counter, and normal inputs)	Not supported
	High-speed counter inputs	Not supported		Not supported		2 max. (high-speed counters 0 and 1)	Not supported
	Normal outputs	Not supported		Not supported		6	Not supported
	Pulse outputs	Not supported		Not supported		2	Not supported
	PWM outputs	Not supported		Not supported		CJ1M-CPU22/23: 2 CJ1M-CPU21: 1	Not supported
	Origin searches/ origin returns	Not supported		Not supported		2	Not supported

A-8 Functions Supported for Unit Versions

Functions Supported for Unit Version 1.3 or Later

CX-Programmer version 9.1 or higher must be used to enable using the functions added for unit version 1.3.

Units		CJ2H CPU Units			
Models		CJ2H-CPU6□-EIP and CJ2H-CPU6□			
Function	Unit version	Unit version 1.3 or later	Unit version 1.2	Unit version 1.1	Unit version 1.0
Special instructions for certain Special I/O Units	CJ1W-NC281/NC481/NC881 Position Control Units: PCU HIGH-SPEED POSITIONING (NCDMV(218))	Supported.	Not supported.	Not supported.	Not supported.
	CJ1W-NC281/NC481/NC881 Position Control Units: PCU POSITIONING TRIGGER (NCDTR(219))	Supported.	Not supported.	Not supported.	Not supported.
New special instructions	SIGNED AREA RANGE COMPARE: ZCPS(088)	Supported.	Not supported.	Not supported.	Not supported.
	DOUBLE SIGNED AREA RANGE COMPARE: ZCPSL(116)	Supported.	Not supported.	Not supported.	Not supported.

Unit Version 1.2 or Later

CX-Programmer version 8.3 or higher must be used to enable using the functions added for unit version 1.2.

Unit	CJ2H CPU Units	
Model	CJ2H-CPU6□-EIP CJ2H-CPU6□	
Unit version	Unit version 1.2 or later	Other unit versions
EM Area Force-setting/resetting	Supported.	Not supported.

Note User programs that use functions of CJ2H CPU Units with unit version 1.2 or later cannot be used with CJ2H CPU Units with unit version 1.1 or earlier. If an attempt is made to transfer a program that uses any of these functions from the CX-Programmer to a CPU Unit with unit version 1.1 or earlier, an error will be displayed and it will not be possible to download to the CPU Unit.

Functions Supported for Unit Version 1.1 or Later

CX-Programmer version 8.1* or higher must be used to enable using the functions added for unit version 1.1.

Units	CJ2H CPU Unit	
Models	CJ2H-CPU6□-EIP and CJ2H-CPU6□	
Function / Unit version	Unit version 1.1	Unit version 1.0
High-speed interrupt function Reduced overhead time for interrupt tasks Minimum interval for scheduled interrupt task execution: 0.1 ms	Supported.	Not supported.
Changing the minimum cycle time setting during operation	Supported.	Not supported.
Synchronous unit operation	Supported.	Not supported.
Addition of immediate refreshing instructions for certain Special I/O Units CJ1W-AD042 High-speed Analog Input Unit: ANALOG INPUT DIRECT CONVERSION (AIDC(216)) CJ1W-DA042V High-speed Analog Output Unit: ANALOG OUTPUT DIRECT CONVERSION (AODC(217))	Supported.	Not supported.
High-speed Serial Communications Units CJ1W-SCU22/SCU32/SCU42 High-speed Serial Communications Units: DIRECT RECEIVE VIA SERIAL COMMUNICATIONS UNIT (DRXDU(261)) and DIRECT TRANSMIT VIA SERIAL COMMUNICATIONS UNIT (DTXDU(262)	Supported.	Not supported.

* CX-Programmer version 8.02 or higher is required to use the high-speed interrupt function and the function to change the minimum cycle time setting during operation.

Note User programs that use functions of CJ2H CPU Units with unit version 1.1 or later cannot be used with CJ2H CPU Units with unit version 1.0 or earlier. If an attempt is made to transfer a program that uses any of these functions from the CX-Programmer to a CPU Unit with unit version 1.0, an error will be displayed and it will not be possible to download to the CPU Unit. If a program file (extension: .OBJ) that uses any of these functions is transferred to a CPU Unit with unit version 1.0, a program error will occur when operation starts or when the function starts and operation of the CPU Unit will stop.

Index

A

address offset 5-92
automatic address allocation 5-59, 6-27
automatic backup 10-39
automatic online connection 11-3, 11-11
automatic transfer at startup 7-11, 10-27
Auxiliary Area 6-2, 6-22, A-106

B

background execution 10-10
background execution settings 9-12
backup memory 10-39
backup restore operation 7-12
backup servicing 3-5
Basic I/O Unit rack response time 9-21
battery-free operation 9-9
block program section 5-108
built-in flash memory 2-2
built-in RAM 2-2

C

checking programs 5-96
CIO Area 6-2, 6-8
  CPU Bus Unit Area 6-2
  Data Link Area 6-2
  DeviceNet Area 6-2
  I/O Area 6-2
  Internal I/O Area 6-2
  Special I/O Unit Area 6-2
clock functions 10-3
Clock Pulses 6-3, 6-44
comment file 7-9
comment memory 10-42
comments 2-3
Comms Instructions Settings in FB 9-13
communications port servicing 3-5
Communications Settings 9-23
Condition Flags 5-103, 6-3, 6-42
Constant Cycle Time 9-15
Counter Area 6-3, 6-33
  Counter Completion Flag Area 6-3
  Counter Completion Flags 6-33
  Counter PV Area 6-3
  Counter PVs 6-33
CPU Bus Unit Area 6-2, 6-15
CPU Bus Unit setup 2-3
CPU Bus Unit Setup Area 4-6, 4-16
CPU Bus Units
  I/O allocations 6-15, 6-26
CPU processing mode 9-30
CPU Unit settings 9-8
current bank 6-29
cycle time 3-4
  computing the cycle time 12-4
  monitoring the cycle time 12-2
cycle time extension 12-13
cyclic refresh 5-79
cyclic tasks 5-14

D

data file 7-8
data format 5-75
Data Link Area 6-2, 6-13
Data Memory Area 6-2, 6-24
Data Registers 6-3, 6-40
data structure 5-56
data type 5-54
debugging 10-63
debugging with simulator 5-97
DeviceNet Area 6-2, 6-18
DeviceNet Comms Instructions in FB 9-13
differential monitoring 10-64
DIP switch 4-2
disable SIOU cyclic refresh 9-19
disabling power OFF interrupts 10-25
Do not Detect Low Battery 9-8

E

EM file memory 7-2
EM file memory functions 10-41
enabling and disabling program overwriting 10-54
enabling/disabling saving to Memory Cards 10-53
error
  Error Flag ON error 5-100
  illegal access error 5-100
  illegal instruction error 5-101
  instruction processing error 5-100
  P_AER Flag ON error 5-100
  program error 5-101
  user program area overflow error 5-101
error simulation function 5-99
event servicing for CPU Bus Units 3-5
event servicing for Special I/O Units 3-5
Execute Process 9-8
Execution Setting 9-7
Extended Data Memory Area 6-2, 6-27
external interrupt task 5-27

F

failure alarm instructions 10-74
FAL(006) instruction 10-74
FALS(007) instruction 10-74
FB Program Area 2-3
file access servicing 3-5
file memory 6-28, 7-2
  comment file 7-9
  data file 7-8
  parameter file 7-8
  program index file 7-9
  program/network symbol file 7-7
  symbols table file 7-9
  unit backup file 7-9

files automatically transferred at startup 7-13
FINS Protection 9-31
flags related to tasks 5-31
forced set/reset 10-63
Forced Status Hold Bit Startup Hold Setting 9-5
FOR-NEXT loop 5-108
FPD(269) instruction 10-76
Free Running Timer 10-6
function blocks 5-40

G

general-purpose files 7-15
global symbols 5-46, 5-48

H

high-speed input 10-9
High-speed interrupt function 10-19
Holding Area 6-2, 6-20
holding settings for operating mode changes and startup .. 10-22
Host Link System 11-16

I

I/O allocation 8-2
  automatic allocation 8-5
  manual allocation 8-9
I/O Area 6-2
I/O interrupt task 5-25
I/O Memory Areas 2-3
I/O memory areas 6-2
I/O refresh times 12-7
I/O refresh timing 5-79
I/O refreshing 3-6, 6-8, 6-10
I/O response time 12-13
I/O table 8-3
  automatic allocation 4-10
  manual allocation 4-10
I/O tables 2-3, 4-5
IL - ILC section 5-108
illegal access errors 5-100
illegal instruction error 5-101
immediate refreshing 5-80, 6-9, 6-11
Index Registers 5-84, 6-3, 6-35
initializing file memory 7-3
input bits 6-8
instruction execution conditions 5-63
instruction execution times A-78
instruction functions A-3
instruction processing error 5-100
instruction variations 5-63
integrated simulation 5-98
Internal I/O Area 6-2
interrupt overhead time 5-21
interrupt task 5-20
interrupt tasks
  external interrupt task 5-20
  extra cyclic task 5-20
  I/O interrupt task 5-20
  power OFF interrupt task 5-20
  scheduled interrupt task 5-20
IOM Hold Bit Startup Hold Setting 9-6
IORF(097) refreshing 6-9, 6-11

J

JMP0 - JME0 section 5-108

L

ladder diagram 5-6
Link Area 6-13
local network table 4-12
local symbols 5-46, 5-48

M

maximum cycle time 10-8
memory areas
  I/O Memory Areas 2-3
  Network Symbols (Tags) 2-3
  Parameter Area 2-3
    CPU Bus Unit Setup 2-3
    I/O Tables 2-3
    PLC name 2-3
    PLC Setup 2-3
    Routing Tables 2-3
  Source and Comment Areas 2-3
    Comments 2-3
    Program Index 2-3
    Source Code 2-3
    Symbol Table 2-3
  User Program Area 2-3
Memory Card 7-2, 7-5
Memory Card recognition time 7-5
memory management functions 10-39
memory map of PLC memory addresses A-141
minimum cycle time 10-7
MONITOR mode 3-8
monitoring the cycle time 10-9

N

network symbols 5-49
network symbols (tags) 2-3
no-protocol communications system 11-17
NT Link System 11-18
number of steps A-78

O

online editing 10-65
online editing servicing 3-5
operating mode 3-8
operating modes 9-7
  MONITOR mode 3-8
  PROGRAM mode 3-8
  RUN mode 3-8
Operation End Time 10-6
operation for power interruptions A-143
Operation Start Time 10-6
output bits 6-10
overseeing 3-4

P

P_AER Flag ON error 5-100
Parameter Area 2-3
Parameter Date 10-5
parameter file 7-8
P_ER Flag ON error 5-100
Peripheral Services 9-30
peripheral servicing 3-5
  backup servicing 3-5
  communications port servicing 3-5
  event servicing for CPU Bus Units 3-5
  event servicing for Special I/O Units 3-5
  file access servicing 3-5
  online editing servicing 3-5
  peripheral USB port service 3-5
  serial port service 3-5
PLC Backup Tool 7-14
PLC name 2-3, 4-4, 10-60
PLC Setup 4-4, 4-8, 9-2
  background execution settings 9-12
  Basic I/O Unit rack response times 9-21
  Comms Instructions Settings in FB 9-13
  Communications Settings 9-23
  CPU processing mode 9-30
  CPU Unit Settings 9-8
  cycle time 9-15
  Disable SIOU Cyclic Refresh 9-19
  Execute Process 9-8
  Execution Setting 9-7
  FINS Protection 9-31
  Forced Status Hold Bit Startup Hold Setting 9-5
  IOM Hold Bit Startup Hold Setting 9-6
  Operating Mode 9-7
  Peripheral Services 9-30
  power OFF detection delay time 9-17
  power OFF interrupt disabled 9-18
  rack response times 9-21
  Scheduled Interrupt Interval 9-16
  Serial Port Settings 9-23
  Set Time to All Events 9-30
  Settings for FINS write protection via network 9-31
  Special I/O Unit cyclic refreshing 9-19
  Startup Hold settings 9-5
  Startup operation settings 9-5
  time and interrupt settings 9-14
  Watch Cycle Time 9-14
PLC setup 2-3
Power Interruption Time 10-5
power OFF detection delay setting 10-24
power OFF detection delay time 9-17
Power OFF Interrupt disabled 9-18
power OFF interrupt task 5-21, 9-18
Power ON Clock Data 10-4
processing at power interruptions 3-7
program capacity 5-5
program error 5-101
program execution 3-4
program index 2-3
program index file 7-9
PROGRAM mode 3-8
program operation protection using production lot numbers 10-55
program section
  block program section 5-108
  FOR-NEXT loop 5-108
  IL - ILC section 5-108
  JMP0 - JME0 section 5-108
  step ladder section 5-108
  subroutine 5-108
program/network symbol file 7-7
programming language 5-4
protocol macros 11-20

R

read protection using passwords 10-50
relay network table 4-12
replacing the entire program 10-43
response performance of Serial PLC Links 12-15
response time for built-in input interrupts 12-14
Retry Counts 9-13
rotary switch 4-2
routing tables 2-3, 4-5, 4-11
RUN mode 3-8
RUN output 10-26

S

safety precautions 1-21
Scheduled Interrupt Interval 9-16
scheduled interrupt task 5-23
sections 5-38
security functions 10-50
  read protection using passwords 10-50
  write protection from FINS commands 10-56
  write-protection using the DIP switch 10-50
self-maintaining bits 6-20
Serial PLC Link 11-21
Serial PLC Link Area 6-2, 6-17
serial port settings 9-23
set time to all events 9-30
setting allocated DM area words for CPU Bus Units ... 4-15
setting allocated DM area words for Special I/O Units 4-15
setting CPU Bus Units and Special I/O Units 8-20
settings for FINS write protection via network 9-31
SFC language 5-9
simple backup file 7-14
simple backup operation 7-11
simulating system errors 10-75
Slot Start Address Settings 8-12
Source and Comment Areas 2-3
source code 2-3
Special I/O Unit Area 6-2, 6-16
Special I/O Unit cyclic refreshing 9-19
Special I/O Units
  I/O allocations 6-26
special program sections 5-108
specifying arrays 5-55
ST language 5-8
Startup Hold settings 9-5
startup initialization 3-2
Startup operation settings 9-5
startup settings and maintenance 10-22
step ladder section 5-108
storing the stop position at errors 10-73
structure of instructions
  flags 5-61

instruction conditions 5-61
operands 5-62
power flow 5-60
subroutine 5-108
symbol table 2-3
symbol tables 5-45
symbols 5-45
symbols table file 7-9
Synchronous Data Refresh Area 6-14
synchronous operation cycle 10-79
synchronous unit operation 10-78

T

tags 5-49
Task Control Instructions 5-17
Task Flag Area 6-3
Task Flags 6-34
tasks 5-11
Temporary Relay Area 6-2, 6-23
test input 10-64
time and interrupt settings 9-14
Time Area
Timer Completion Flag Area 6-2
Timer Area 6-2, 6-31
Timer PV Area 6-2
times stored in memory
Operation Start Time and Operation End Time 10-6
Power Interruption Time 10-5
Power ON Clock Data 10-4
Total Power ON Time 10-5
User Program and Parameter Date 10-5
Total Power ON Time 10-5
trace memory 6-28
tracing data 10-68
turning OFF outputs 10-67

U

unit backup file 7-9
unit management functions 10-35
User Program Area 2-3
user program area overflow error 5-101
User Program Date 10-5
user-defined data type 5-56

V

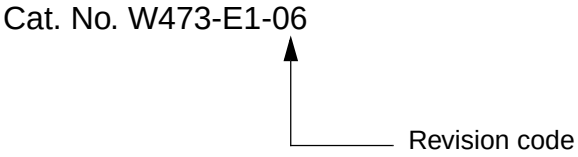
variation 6-9

W

Watch Cycle Time 9-14
Work Area 6-2, 6-19
write protection from FINS commands 10-56
write-protection using the DIP switch 10-50

Revision History

A manual revision code appears as a suffix to the catalog number on the front cover of the manual.



The following table outlines the changes made to the manual during each revision. Page numbers refer to the previous version.

Revision code	Date	Revised content
01	July 2008	Original production
02	December 2008	Added the CJ2H-CPU6□. Added information on unit version 1.1 of the CJ2 CPU Units. Added the following Basic I/O Units: CJ1W-ID212, CJ1W-ID233, CJ1W-OD213, and CJ1W-OD234.
03	February 2009	Added information on the synchronous unit operation function. Added the following Position Control Units: CJ1W-NC214, CJ1W-NC234, CJ1W-NC414, and CJ1W-NC434.
04	July 2009	Added the CJ1W-AD042 Analog Input Unit. Added the CJ1W-DA042V Analog Output Unit.
05	October 2009	Added the CJ1W-SCU22/32/42 Serial Communications Units. Added the DRXDU(261) and DTXDU(262) instructions. Added information on the EM Area force-setting/resetting function (a new unit version 1.2 function).
06	February 2010	Added the CJ2M-CPU□□.

OMRON Corporation **Industrial Automation Company**
Tokyo, JAPAN
Contact: www.ia.omron.com

Regional Headquarters
OMRON EUROPE B.V.
Wegalaan 67-69-2132 JD Hoofddorp
The Netherlands
Tel: (31)2356-81-300/Fax: (31)2356-81-388

OMRON ELECTRONICS LLC
One Commerce Drive Schaumburg,
IL 60173-5302 U.S.A.
Tel: (1) 847-843-7900/Fax: (1) 847-843-7787

OMRON ASIA PACIFIC PTE. LTD.
No. 438A Alexandra Road # 05-05/08 (Lobby 2),
Alexandra Technopark,
Singapore 119967
Tel: (65) 6835-3011/Fax: (65) 6835-2711

OMRON (CHINA) CO., LTD.
Room 2211, Bank of China Tower,
200 Yin Cheng Zhong Road,
PuDong New Area, Shanghai, 200120, China
Tel: (86) 21-5037-2222/Fax: (86) 21-5037-2200

Authorized Distributor:

© OMRON Corporation 2009 All Rights Reserved.
In the interest of product improvement,
specifications are subject to change without notice.

Printed in Japan
Cat. No. W473-E1-06 0210